

DATA TRANSFER INSTRUCTIONS

Opcode	Operand	Description
Copy from source to destination		
MOV	Rd, Rs	This instruction copies the contents of the source register into the destination register; the contents of the source register are not altered. If one of the operands is a memory location, its location is specified by the contents of the HL registers. Example: MOV B, C or MOV B, M
M, Rs		
Rd, M	Move immediate 8-bit	The 8-bit data is stored in the destination register or memory. If the operand is a memory location, its location is specified by the contents of the HL registers. Example: MVI B, 57H or MVI M, 57H The contents of a memory location, specified by a 16-bit address in the operand, are copied to the accumulator. The contents of the source are not altered. Example: LDA 2034H The contents of the designated register pair point to a memory location. This instruction copies the contents of that memory location into the accumulator. The contents of either the register pair or the memory location are not altered. Example: LDAX B
MVI	Rd, data	
M, data		The instruction loads 16-bit data in the register pair designated in the operand. Example: LXI H, 2034H or LXI H, XYZ
LDA	16-bit address	
	Load accumulator indirect	The instruction copies the contents of the memory location pointed out by the 16-bit address into register L and copies the contents of the next memory location into register H. The contents of source memory locations are not altered. Example: LHLD 2040H
LDAX	B/D Reg. pair	
	Load register pair immediate	The contents of the accumulator are copied into the memory location specified by the operand. This is a 3-byte instruction, the second byte specifies the low-order address and the third byte specifies the high-order address. Example: STA 4350H
LXI	Reg. pair, 16-bit data	
	Load H and L registers direct	The contents of the accumulator are copied into the memory location specified by the contents of the operand (register pair). The contents of the accumulator are not altered. Example: STAX B
LHLD	16-bit address	
	Store accumulator direct	The contents of register L are stored into the memory location specified by the 16-bit address in the operand and the contents of H register are stored into the next memory location by incrementing the operand. The contents of registers HL are not altered. This is a 3-byte instruction, the second byte specifies the low-order address and the third byte specifies the high-order address. Example: SHLD 2470H
STA	16-bit address	
	Store accumulator indirect	The contents of register H are exchanged with the contents of register D, and the contents of register L are exchanged with the contents of register E. Example: XCHG
STAX	Reg. pair	
	Store H and L registers direct	Copy H and L registers to the stack pointer
SHLD	16-bit address	
	Exchange H and L with D and E	The instruction loads the contents of the H and L registers into the stack pointer register, the contents of the H register provide the high-order address and the contents of the L register provide the low-order address. The contents of the H and L registers are not altered. Example: SPHL
XCHG	none	
	Exchange H and L with top of stack	

XTHL none The contents of the L register are exchanged with the stack location pointed out by the contents of the stack pointer register. The contents of the H register are exchanged with the next stack location (SP+1); however, the contents of the stack pointer register are not altered.

Example: XTHL

Push register pair onto stack

PUSH Reg. pair

Pop off stack to register pair

The contents of the register pair designated in the operand are copied onto the stack in the following sequence. The stack pointer register is decremented and the contents of the highorder register (B, D, H, A) are copied into that location. The stack pointer register is decremented again and the contents of the low-order register (C, E, L, flags) are copied to that location.

Example: PUSH B or PUSH A

POP Reg. pair

The contents of the memory location pointed out by the stack pointer register are copied to the low-order register (C, E, L, status flags) of the operand. The stack pointer is incremented by 1 and the contents of that memory location are copied to the high-order register (B, D, H, A) of the operand. The stack pointer register is again incremented by 1. Example: POP H or POP A

Output data from accumulator to a port with 8-bit address

OUT 8-bit port address The contents of the accumulator are copied into the I/O port specified by the operand.

Example: OUT F8H

Input data to accumulator from a port with 8-bit address

IN 8-bit port address The contents of the input port designated in the operand are read and loaded into the accumulator. Example: IN 8CH

ARITHMETIC INSTRUCTIONS

Opcode	Operand	Description
--------	---------	-------------

Add register or memory to accumulator

ADD	R	The contents of the operand (register or memory) are
------------	---	--

M added to the contents of the accumulator and the result is stored in the accumulator. If the operand is a memory location, its location is specified by the contents of the HL registers. All flags are modified to reflect the result of the addition.

Example: ADD B or ADD M

Add register to accumulator with carry

ADC	R	The contents of the operand (register or memory) and
------------	---	--

M the Carry flag are added to the contents of the accumulator

and the result is stored in the accumulator. If the operand is a memory location, its location is specified by the contents of the HL registers. All flags are modified to reflect the result of the addition.

Example: ADC B or ADC M

Add immediate to accumulator

ADI	8-bit data	The 8-bit data (operand) is added to the contents of the
------------	------------	--

accumulator and the result is stored in the accumulator. All flags are modified to reflect the result of the addition.

Example: ADI 45H

Add immediate to accumulator with carry

ACI 8-bit data The 8-bit data (operand) and the Carry flag are added to the contents of the accumulator and the result is stored in the accumulator. All flags are modified to reflect the result of the addition.

Example: ACI 45H

Add register pair to H and L registers

DAD Reg. pair The 16-bit contents of the specified register pair are added to the contents of the HL register and the sum is stored in the HL register. The contents of the source register pair are not altered. If the result is larger than 16 bits, the CY flag is set.

No other flags are affected. Example: DAD H

Subtract register or memory from accumulator

SUB R The contents of the operand (register or memory) are

M subtracted from the contents of the accumulator, and the result is stored in the accumulator. If the operand is a memory location, its location is specified by the contents of the HL registers. All flags are modified to reflect the result of the subtraction.

Example: SUB B or SUB M

Subtract source and borrow from accumulator

SBB R The contents of the operand (register or memory) and

M the Borrow flag are subtracted from the contents of the accumulator and the result is placed in the accumulator. If the operand is a memory location, its location is specified by the contents of the HL registers. All flags are modified to reflect the result of the subtraction. Example: SBB B or SBB M

Subtract immediate from accumulator

SUI 8-bit data The 8-bit data (operand) is subtracted from the contents of the accumulator and the result is stored in the accumulator. All flags are modified to reflect the result of the subtraction.

Example: SUI 45H

Subtract immediate from accumulator with borrow

SBI 8-bit data The 8-bit data (operand) and the Borrow flag are subtracted from the contents of the accumulator and the result is stored in the accumulator. All flags are modified to reflect the result of the subtraction.

Example: SBI 45H

Increment register or memory by 1

INR R The contents of the designated register or memory) are

M incremented by 1 and the result is stored in the same place. If

the operand is a memory location, its location is specified by the contents of the HL registers.

Example: INR B or INR M

Increment register pair by 1

INX **R** The contents of the designated register pair are incremented

by 1 and the result is stored in the same place. Example: **INX H**

Decrement register or memory by 1

DCR **R** The contents of the designated register or memory are

M decremented by 1 and the result is stored in the same place. If

the operand is a memory location, its location is specified by

the contents of the HL registers.

Example: **DCR B** or **DCR M**

Decrement register pair by 1

DCX **R** The contents of the designated register pair are decremented

by 1 and the result is stored in the same place.

Example: **DCX H**

Decimal adjust accumulator

DAA none The contents of the accumulator are changed from a binary

value to two 4-bit binary coded decimal (BCD) digits. This is

the only instruction that uses the auxiliary flag to perform the

binary to BCD conversion, and the conversion procedure is

described below. S, Z, AC, P, CY flags are altered to reflect

the results of the operation.

If the value of the low-order 4-bits in the accumulator is greater than 9 or if AC flag is set, the instruction adds 6 to the low-order four bits.

If the value of the high-order 4-bits in the accumulator is greater than 9 or if the Carry flag is set, the instruction adds 6 to the high-order four bits.

Example: **DAA**

BRANCHING INSTRUCTIONS

Opcode	Operand	Description
Jump unconditionally		
JMP	16-bit address	The program sequence is transferred to the memory location specified by the 16-bit address given in the operand. Example: JMP 2034H or JMP XYZ
Jump conditionally		
Operand: 16-bit address		
The program sequence is transferred to the memory location specified by the 16-bit address given in the operand based on the specified flag of the PSW as described below. Example: JZ 2034H or JZ XYZ		
Opcode	Description	Flag Status
JC	Jump on Carry	CY = 1

	JNC	Jump on no Carry	CY = 0
JP		Jump on positive	S = 0
JM		Jump on minus	S = 1
JZ		Jump on zero	Z = 1
JNZ		Jump on no zero	Z = 0
JPE		Jump on parity even	P = 1
JPO		Jump on parity odd	P = 0

Unconditional subroutine call

CALL 16-bit address

Call conditionally

Operand: 16-bit address

The program sequence is transferred to the memory location specified by the 16-bit address given in the operand. Before the transfer, the address of the next instruction after CALL (the contents of the program counter) is pushed onto the stack.

Example: CALL 2034H or CALL XYZ

The program sequence is transferred to the memory location specified by the 16-bit address given in the operand based on the specified flag of the PSW as described below. Before the transfer, the address of the next instruction after the call (the contents of the program counter) is pushed onto the stack. Example: CZ 2034H or CZ XYZ

Opcode	Description	Flag Status
CC	Call on Carry	CY = 1
CNC	Call on no Carry	CY = 0
CP	Call on positive	S = 0
CM	Call on minus	S = 1
CZ	Call on zero	Z = 1
CNZ	Call on no zero	Z = 0
CPE	Call on parity even	P = 1
CPO	Call on parity odd	P = 0

Return from subroutine unconditionally

RET none The program sequence is transferred from the subroutine to the calling program. The two bytes from the top of the stack are copied into the program counter, and program execution begins at the new address. Example: RET

Return from subroutine conditionally

Operand: none

The program sequence is transferred from the subroutine to the calling program based on the specified flag of the PSW as described below. The two bytes from the top of the stack are copied into the program counter, and program execution begins at the new address. Example: RZ

Opcode	Description	Flag Status
RC	Return on Carry	CY = 1
RNC	Return on no Carry	CY = 0
RP	Return on positive	S = 0
RM	Return on minus	S = 1
RZ	Return on zero	Z = 1
RNZ	Return on no zero	Z = 0
RPE	Return on parity even	P = 1
RPO	Return on parity odd	P = 0

Load program counter with HL contents

PCHL none

Restart

The contents of registers H and L are copied into the program counter. The contents of H are placed as the high-order byte and the contents of L as the low-order byte. Example: PCHL

RST 0-7

The RST instruction is equivalent to a 1-byte call instruction to one of eight memory locations depending upon the number. The instructions are generally used in conjunction with interrupts and inserted using external hardware. However these can be used as software instructions in a program to transfer program execution to one of the eight locations. The addresses are:

Instruction	Restart Address
RST 0	0000H
RST 1	0008H
RST 2	0010H
RST 3	0018H
RST 4	0020H
RST 5	0028H
RST 6	0030H
RST 7	0038H

The 8085 has four additional interrupts and these interrupts generate RST instructions internally and thus do not require any external hardware.

These instructions and their Restart addresses are:

Interrupt	Restart Address
TRAP	0024H
RST 5.5	002CH
RST 6.5	0034H
RST 7.5	003CH

LOGICAL INSTRUCTIONS

Opcode Operand Description

Compare register or memory with accumulator

CMP R The contents of the operand (register or memory) are
M compared with the contents of the accumulator. Both

contents are preserved. The result of the comparison is shown by setting the flags of the PSW as follows: if (A) < (reg/mem): carry flag is set if (A) = (reg/mem): zero flag is set

if (A) > (reg/mem): carry and zero flags are reset

Example: CMP B or CMP M

Compare immediate with accumulator

CPI 8-bit data The second byte (8-bit data) is compared with the contents of the accumulator. The values being compared remain unchanged. The result of the comparison is shown by setting the flags of the PSW as follows: if (A) < data: carry flag is set if (A) = data: zero flag is set

if (A) > data: carry and zero flags are reset

Example: CPI 89H

Logical AND register or memory with accumulator

ANA R The contents of the accumulator are logically ANDed with

M the contents of the operand (register or memory), and the result is placed in the accumulator. If the operand is a memory location, its address is specified by the contents of HL registers. S, Z, P are modified to reflect the result of the operation. CY is reset. AC is set.

Example: ANA B or ANA M

Logical AND immediate with accumulator

ANI 8-bit data The contents of the accumulator are logically ANDed with the 8-bit data (operand) and the result is placed in the accumulator. S, Z, P are modified to reflect the result of the operation. CY is reset. AC is set.

Example: ANI 86H

Exclusive OR register or memory with accumulator

XRA R The contents of the accumulator are Exclusive ORed with

M the contents of the operand (register or memory), and the result is placed in the accumulator. If the operand is a memory location, its address is specified by the contents of HL registers. S, Z, P are modified to reflect the result of the operation. CY and AC are reset. Example: XRA B or XRA M

Exclusive OR immediate with accumulator

XRI 8-bit data The contents of the accumulator are Exclusive ORed with the 8-bit data (operand) and the result is placed in the accumulator. S, Z, P are modified to reflect the result of the operation. CY and AC are reset.

Example: XRI 86H

Logical OR register or memory with accumulator

ORA R The contents of the accumulator are logically ORed with

M the contents of the operand (register or memory), and the result is placed in the accumulator. If the operand is a memory location, its address is specified by the contents of HL registers. S, Z, P are modified to reflect the result of the operation. CY and AC are reset. Example: ORA B or ORA M

Logical OR immediate with accumulator

ORI 8-bit data The contents of the accumulator are logically ORed with the 8-bit data (operand) and the result is placed in the accumulator. S, Z, P are modified to reflect the result of the operation. CY and AC are reset. Example: ORI 86H

RLC none Each binary bit of the accumulator is rotated left by one position. Bit D7 is placed in the position of D0 as well as in the Carry flag. CY is modified according to bit D7. S, Z, P, AC are not affected. Example: RLC

RRC none Each binary bit of the accumulator is rotated right by one position. Bit D0 is placed in the position of D7 as well as in the Carry flag. CY is modified according to bit D0. S, Z, P, AC are not affected. Example: RRC

Rotate accumulator left through carry

RAL none Each binary bit of the accumulator is rotated left by one

position through the Carry flag. Bit D7 is placed in the Carry flag, and the Carry flag is placed in the least significant position D0. CY is modified according to bit D7. S, Z, P, AC are not affected. Example: RAL

Rotate accumulator right through carry

RAR none Each binary bit of the accumulator is rotated right by one position through the Carry flag. Bit D0 is placed in the Carry flag, and the Carry flag is placed in the most significant position D7. CY is modified according to bit D0. S, Z, P, AC are not affected. Example: RAR

CMA none The contents of the accumulator are complemented. No flags are affected. Example: CMA

CMC none
 Set Carry
 STC none

The Carry flag is complemented. No other flags are affected. Example: CMC
 The Carry flag is set to 1. No other flags are affected. Example: STC

CONTROL INSTRUCTIONS

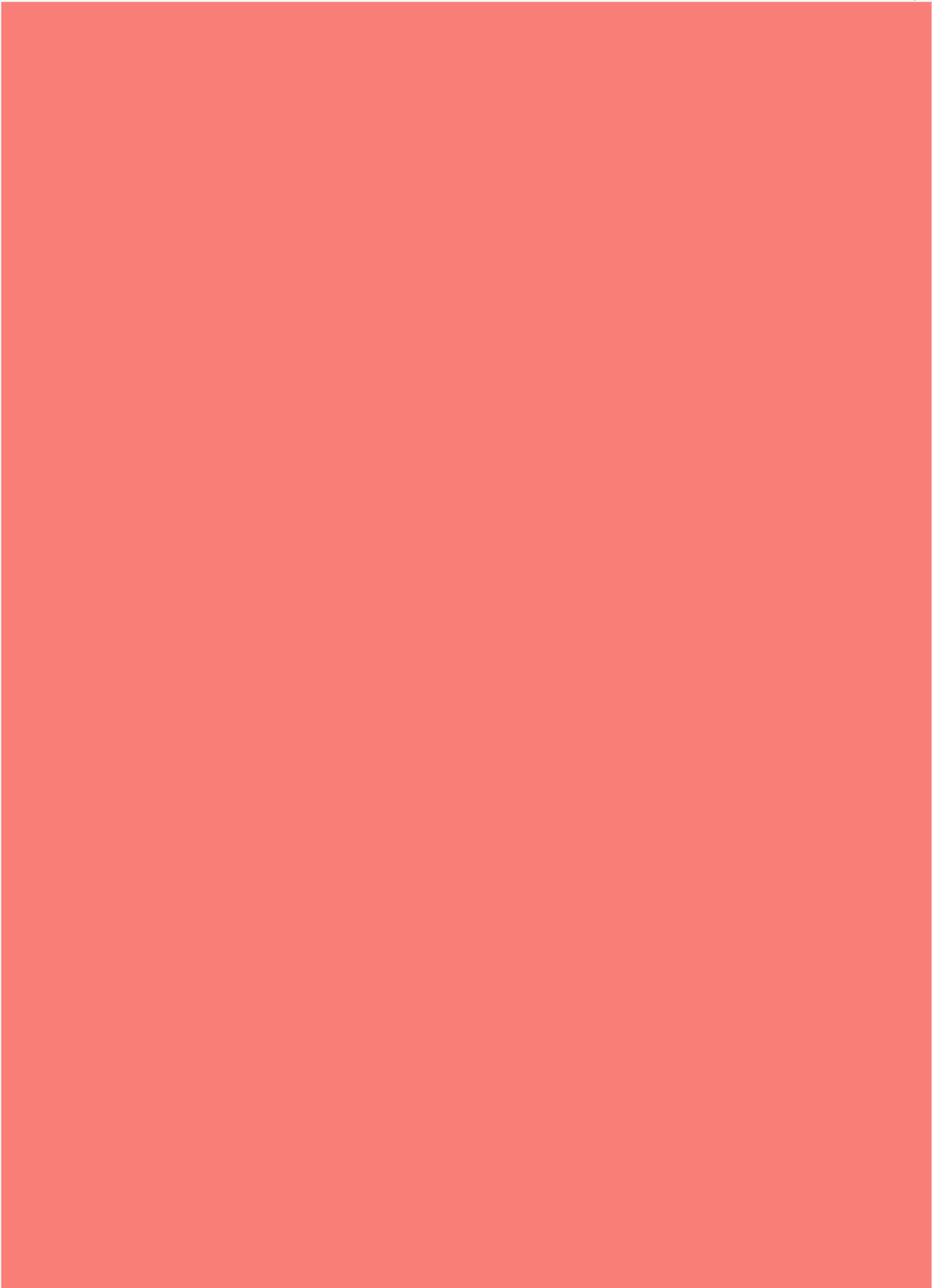
Opcode	Operand	Description
No operation		
NOP	none	No operation is performed. The instruction is fetched and decoded. However no operation is executed. Example: NOP
HLT	none	The CPU finishes executing the current instruction and halts any further execution. An interrupt or reset is necessary to exit from the halt state. Example: HLT
DI	none	The interrupt enable flip-flop is reset and all the interrupts except the TRAP are disabled. No flags are affected. Example: DI
EI	none	The interrupt enable flip-flop is set and all interrupts are enabled. No flags are affected. After a system reset or the acknowledgement of an interrupt, the interrupt enable flipflop is reset, thus disabling the interrupts. This instruction is necessary to reenale the interrupts (except TRAP). Example: EI
Read interrupt mask		
RIM	none	This is a multipurpose instruction used to read the status of interrupts 7.5, 6.5, 5.5 and read serial data input bit. The instruction loads eight bits in the accumulator with the following interpretations. Example: RIM
Set interrupt mask		
SIM	none	This is a multipurpose instruction and used to implement the 8085 interrupts 7.5, 6.5, 5.5, and serial data output. The instruction interprets the accumulator contents as follows. Example: SIM

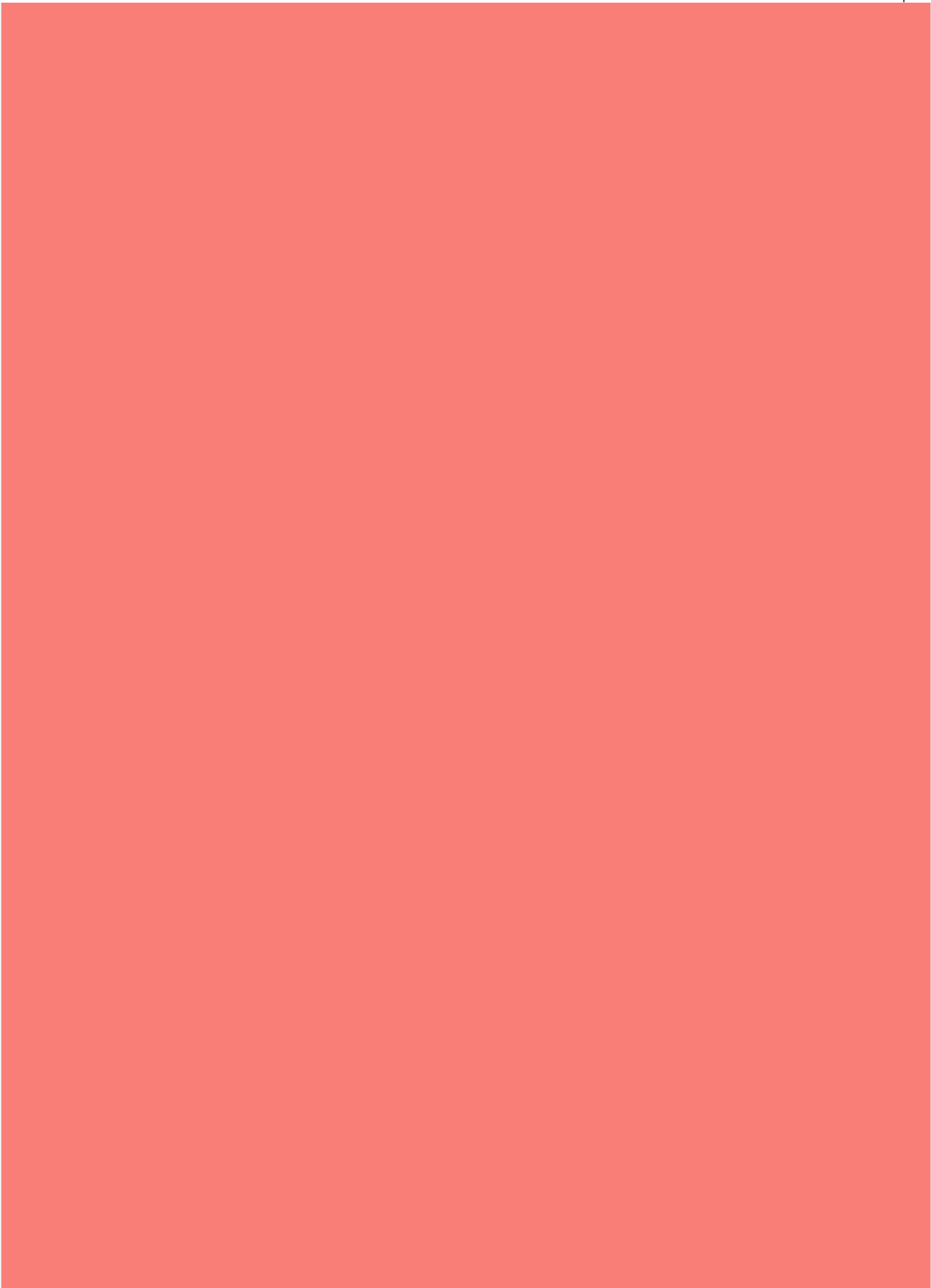


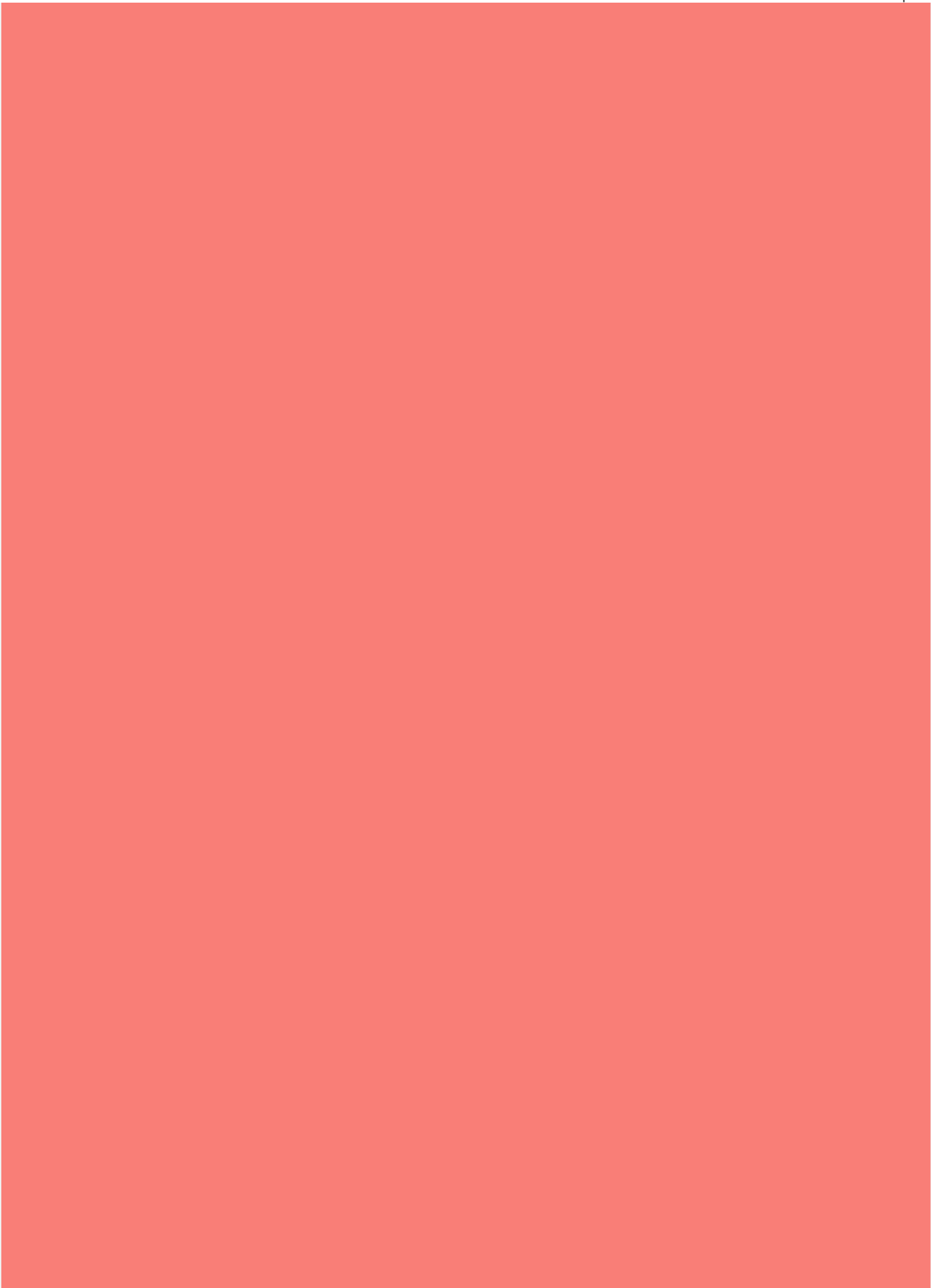
Manas Sharma

I'm a physicist specializing in computational material science with a PhD in Physics from Friedrich-Schiller University Jena, Germany. I write efficient codes for simulating light-matter interactions at atomic scales. I like to develop Physics, DFT, and Machine Learning related apps and software from time to time. Can code in most of the popular languages. I like to share my knowledge in Physics and applications using this Blog and a YouTube channel.

manas.bragitoff.com/







**Share this:**

[Click to share on Facebook \(Opens in new window\)](#)

[Click to share on Twitter \(Opens in new window\)](#)

[Click to share on WhatsApp \(Opens in new window\)](#)

[Click to share on Pinterest \(Opens in new window\)](#)

[Click to share on Reddit \(Opens in new window\)](#)

[Click to share on LinkedIn \(Opens in new window\)](#)

[Click to email a link to a friend \(Opens in new window\)](#)

[Click to print \(Opens in new window\)](#)

[Click to share on Tumblr \(Opens in new window\)](#)

[Click to share on Pocket \(Opens in new window\)](#)

[Click to share on Telegram \(Opens in new window\)](#)

[wpedon id="7041" align="center"]