MICRO-PROCESSOR AND MICRO-COMPUTERS

INTRODUCTION TO MICROPROCESSOR ARCHITECTURES

A Microprocessor is a multipurpose programmable logic device which reads the binary instructions from a storage device called 'Memory' accepts binary data as input and process data according to the instructions and gives the results as output. So, you can understand the Microprocessor as a programmable digital device, which can be used for both data processing and control applications. In view of a computer student, it is the CPU of a Computer or heart of the computer. A computer which is built around a microprocessor is called a microcomputer. A microcomputer system consists of a CPU (microprocessor), memories (primary and secondary) and I/O devices as shown in the block diagram in Fig 1. The memory and I/O devices are linked by data and address (control) buses. The CPU communicates with only one peripheral at a time by enabling the peripheral by the control signal. For example to send data to the output device, the CPU places the device address on the address bus, data on the data bus and enables the output device. The other peripherals that are not enabled remain in high impedance state called tri-state.

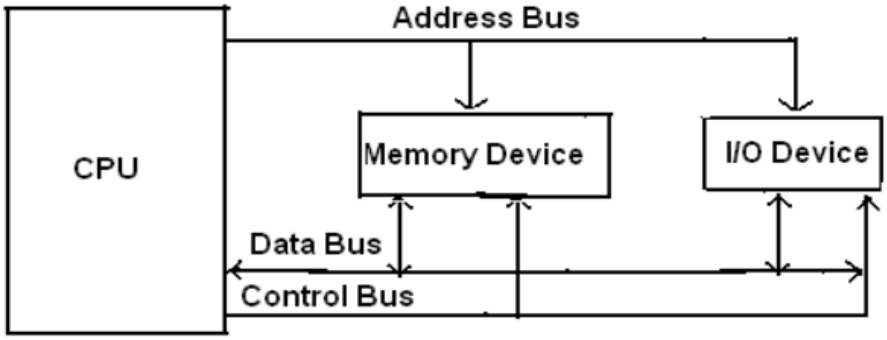

Fig.1 Block diagram of a Microcomputer

Evolution of Microprocessors

The first Microprocessor (4004) was designed by Intel Corporation which was founded by Moore and Noyce in 1968.

In the early years, Intel focused on developing semiconductor memories (DRAMs and EPROMs) for digital computers.

In 1969, a Japanese Calculator manufacturer, Busicom approached Intel with a design for a small calculator which need 12 custom chips. Ted Hoff, an Intel Engineer thought that a general purpose logic device could replace the multiple components.

This idea led to the development of the first so called microprocessor. So, Microprocessors started with a modest beginning of drivers for calculators.

With developments in integration technology Intel was able to integrate the additional chips like 8224 clock generator and the 8228 system controller along with 8080 microprocessor with in a single chip and released the 8 bit microprocessor 8085 in the year 1976. The 8085 microprocessor consisted of 6500 MOS transistors and could work at clock frequencies of 3-5 MHz. It works on a single +5 volts supply. The other improved 8 bit microprocessors include Motorola MC 6809, Zilog Z-80 and RCA COSMAC.

In 1978, Intel introduced the 16 bit microprocessor 8086 and 8088 in 1979. IBM selected the Intel 8088 for their personal computer (IBM-PC).8086 microprocessor made up of 29,000 MOS transistors and could work at a clock speed of 5-10 MHz. It has a 16-bit ALU with 16-bit data bus and 20-bit address bus. It can address up to 1MB of address space. The pipelining concept was used for the first time to improve the speed of the processor. It had a pre-fetch queue of 6 instructions where in the instructions to be executed were fetched during the execution of an instruction. It means 8086 architecture supports parallel processing. The 8088 microprocessor is similar to 8086 processor in architecture ,but the basic difference is it has only 8-bit data bus even though the ALU is of 16-bit.It has a pre-fetch queue of 4-instructions only.

In 1982 Intel released another 16-bit processor called 80186 designed by a team under the leadership of Dave Stamm. This is having higher reliability and faster operational speed but at a lower cost. It had a pre-fetch queue of 6-instructions and it is suitable for high volume applications such as computer workstations, word-processor and personal computers. It is made up of 134,000 MOS transistors and could work at clock rates of 4 and 6 MHz. This is also comes under first generation of Microprocessors.

Intel released another 16 bit microprocessor 80286 having 1, 34,000 transistors in 1981. It was used as CPU in PC-ATs in 1982. It is the second generation microprocessor, more advanced to 80186 processor. It could run at clock speeds of 6 to 12.5 MHz .It has a 16-bit data bus and 24-bit address bus, so that it can address up to 16MB of address space and 1GB of virtual memory. It had a pre-fetch queue of 6 instructions .Intel introduced the concept of protected mode and virtual mode to ensure proper operation. It also had on-chip memory management unit (MMU) .This was popularly called as Intel 286 in those days.

In 1985, Intel released the first 32 bit processor 80386, with 275,000 transistors. It has 32-bit data bus and 32-bit address bus so that it can address up to a total of 4GB memory also a virtual memory space of 64TB.It could process five million instructions per second and could work with all popular operating systems including Windows. It has a pre-fetch queue of length 16-bytes with extensive memory management capabilities. It is incorporated with a concept called paging in addition to segmentation technique. It uses a math co-processor called 80387.

Intel introduced 80486 microprocessor with a built-in maths co-processor and with 1.2 million transistors. It could run at the clock speed of 50 MHz This is also a 32 bit processor but it is twice as fast as 80386.The additional features in 486 processor are the built-in Cache and built-in math co-processors. The address bus here is bidirectional because of presence of cache memory.

On 19th October, 1992, Intel released the Pentium-I Processor with 3.1 million transistors. So, the Pentium began as fifth generation of the Intel x86 architecture. This Pentium was a backward compatible while offering new features. The revolutionary technology followed is that the CPU is able to execute two instruction at the same time. This is known as super scalar technology. The Pentium uses a 32-bit expansion bus, however the data bus is 64 bits.

The 7.5 million transistors based chip, Intel Pentium II processor was released in 1997. It works at a clock speed of 300M.Hz. Pentium II uses the Dynamic Execution Technology which consists of three different facilities namely, Multiple branch prediction, Data flow analysis, and Speculative execution unit. Another important feature is a thermal sensor located on the mother board can monitor the die temperature of the processor. For thermal management applications.

Intel Celeron Processors were introduced in the year 1999. Pentium-III processor with 9.5 million transistors was introduced in 1999. It also uses dynamic execution micro-architecture, a unique combination of multiple branch prediction, dataflow analysis and speculative execution. The Pentium III has improved MMX and processor serial number feature. The improved MMX enables advanced imaging, 3D streaming audio and video, and speech recognition for enhanced Internet facility.

Pentium-IV with 42 million transistors and 1.5 GHz clock speed was released by Intel in November 2000. The Pentium 4 processor has a system bus with 3.2 G-bytes per second of bandwidth. This high bandwidth is a key reason for applications that stream data from memory. This bandwidth is achieved with 64 –bit wide bus capable of transferring data at a rate of 400 MHz. The Pentium 4 processor enables real-time MPEG2 video encoding and near real-time MPEG4 encoding, allowing efficient video editing and video conferencing.

Intel with partner Hewlett-Packard developed the next generation 64-bit processor architecture called IA-64 .This first implementation was named Itanium. Itanium processor which is the first in a family of 64 bit products was introduced in the year 2001.The Itanium processor was specially designed to provide a very high level of parallel processing ,to enable high performance without requiring very high clock frequencies .Key strengths of the Itanium architecture include ,up to 6 instructions/cycle. The Itanium processor can handle up to 6 simultaneous 64 –bit instructions per clock cycle.

The Itanium II is an IA-64 microprocessor developed jointly by Hewlett-Packard (HP) and Intel and released on July 8,2002..It is theoretically capable of performing nearly 8 times more work per clock cycle than other CISC and RISC architectures due to its parallel computing micro-architecture. The recent Itanium processor features a split L2 cache, adding a dedicated 1MB L2 cache for instructions and thereby effectively growing the original 256KBL2 cache, which becomes a dedicated data cache. The first Itanium 2 processor (code named McKinley) was more powerful than the original Itanium processor, with approximately two times performance.

Pentium 4EE was released by Intel in the year 2003 and Pentium 4E was released in the year 2004.

The Pentium Dual-Core brand was used for mainstream X86-architecture microprocessors from Intel from 2006 to 2009 The 64 bit Intel Core2 was released on July 27,2006. In terms of features, price and performance at a given clock frequency, Pentium Dual-Core processors were positioned above Celeron but below Core and Core 2 microprocessors in Intel's product range. The Pentium Dual-Core was also a very popular choice for over clocking, as it can deliver optimal performance (when over clocked) at a low price.

The Pentium Dual Core, which consists of 167 million transistors was released on January 21, 2007. Intel Core Duo consists of two cores on one die, a 2 MB L2 cache shared by both cores, and an arbiter bus that controls both L2 cache and FSB access.

Core 2 Quad processors are multi-chip modules consisting of two dies similar to those used in Core 2 Duo, forming a quad-core processor. While this allows twice the performance to a dual-core processors at the same clock frequency in ideal conditions, this is highly workload specific and requires applications to take advantage of the extra cores.

In September.2009, new Core i7 models based on the Lynnfield desktop quad-core processor and the Clarksfield quad-core mobile were added, and models based on the Arrandale dual-core mobile processor have been announced. The first six-core processor in the Core lineup is the Gulftown, which was launched on March 16, 2010. Both the regular Core i7 and the Extreme Edition are advertised as five stars in the Intel Processor Rating.

ASSEMBLY LANGUAGE PROGRAMMING EXAMPLES:

Addition Programs

Example 1: Addition of two 8-bit numbers whose sum is 8-bits.

Explanation: This assembly language program adds two 8-bit numbers stored in two memory locations .The sum of the two numbers is 8-bits only.The necessary algorithm and flow charts are given below.

ALGORITHM:

Step1. : Initialize H-L pair with memory address XX00 (say: 9000).

Step2. : Clear accumulator.

Step3. : Add contents of memory location M to accumulator.

Step4. : Increment memory pointer (i.e. XX01).

Step5. : Add the contents of memory indicated by memory pointer to accumulator.

Step6. : Store the contents of accumulator in 9002.

Step7. : Halt

PROGRAM:

Address of the memory location	Hex code	Label	Mnemonics		Comments
			Op-code	Operand	

8000	21,00,90		LXI	H, 9000	Initialise memory pointer to point the first data location 9000.
8003	3E		MVI	A, 00	Clear accumulator
8004	00				
8005	86		ADD	A, M	The first number is added to accumulator [A] ← [A] + M
8006	23		INX	H	Increment the memory pointer to next location of the Data.
8007	86		ADD	A, M	The 2nd number is added to contents of accumulator
8008	32		STA	9002	The contents of accumulator are stored in memory location 9002.
8009	02				
800A	90				
800B	76		HLT		Stop the execution

Ex: Input: Ex: (i) 9000 – 29 H **Ex :(ii) 9000 –49 H**

9001 – 16 H **9001 –32 H**

Result: Ex: (i) 9002 – 3F H **Ex :(ii) 9002 – 7B**

Example 2: Addition of two 8-bit numbers whose sum is 16 bits.

Explanation: The first 8-bit number is stored in one memory location (say 8500) and the second 8-bit number is stored in the next location (8501).Add these two numbers and check for carry. Store the LSB of the sum in one memory location (8502) and the MSB (carry) in the other location(8503).

ALGORITHM:

Step1. : Initialize H-L pair with memory address X (say: 8500).

Step2. : Clear accumulator.

Step3. : Add contents of memory location M to accumulator.

Step4. : Increment memory pointer (i.e. 8501).

Step5. : Add the contents of memory indicated by memory pointer to accumulator.

Step6. : Check for Carry

Step 7 : Store the sum in 8502.

Step8 : Store the Carry in 8503 location

Step 9 : Halt

PROGRAM:

Address of the memory location	Hex code	Label	Mnemonics		Comments
			Op-code	Operand	
8000	21,00,85		LXI	H, 8500 H	Initialise memory pointer to point the first data location 9000.
8003	3E		MVI	A,00	Clear accumulator
8004	00				
8005	86		ADD	A, M	The first number is added to accumulator [A]← [A]+M
8006	0E		MVI	C,00	Initial value of Carry is 0
8007	00				
8008	23		INX	H	Increment the memory pointer to next location of the Data.
8009	86		ADD	A, M	The 2nd number is added to contents of accumulator
800A	32		JNC	FWD	Is Carry exists ? No,go to the label FWD
800B	0E				
800C	80				
800D	0C		INR	C	Make carry =1
800E	32	FWD	STA	8502 H	The sum is stored in memory location 8502.
800F	02				
8010	85				
8011	79		MOV	A,C	
8012	32		STA	8503 H	Store the carry at 8503 location

8013	03				
8014	85				
8015	76		HLT		Stop the execution

Ex: Input: Ex : 8500 – 97 H **RESULT: 8502 – 32 H**

8501 – 98H **8503 -- 01 H**

Example 3: Decimal addition of two 8-bit numbers whose sum is 16 bits.

Explanation: Decimal addition of two 8-bit numbers is same as that of two 8-bit numbers program. Except that the use of DAA instruction. The first 8-bit number is stored in one memory location (say 8500) and the second 8-bit number is stored in the next location(8501).Add these two numbers and use the DAA instruction to get the result in decimal. Also check for carry. Store the LSB of the sum in one memory location(8502) and the MSB (carry) in the other location(8503).

ALGORITHM:

Step1. : Initialize H-L pair with memory address XXXX (say: 8500).

Step2. : Clear Carry register C.

Step3. : Move contents of memory location M to accumulator.

Step4. : Increment memory pointer (i.e. 8501).

Step5. : Add the contents of memory indicated by memory pointer to accumulator.

Step6. : Apply the instruction DAA(Decimal adjust after addition)

Step7: Check for Carry

Step8: Store the sum in XX02.

Step9: Store the Carry in XX03 location

Step10: Halt

PROGRAM

Address of the memory location	Hex code	Label	Mnemonics		Comments
			Op-code	Operand	
8000	21, 00,85		LXI	H, 8500 H	Initialise memory pointer to point the first data location 9000.

8003	0E		MVI	C, 00	Clear accumulator
8004	00				
8005	7E		MOV	A, M	The first number is added to accumulator [A]← [A]+M
8006	23		INX	H	Increment the memory pointer to next location of the Data.
8007	86		ADD	A, M	The 2nd number is added to contents of accumulator
8008	27		DAA		
8009	D2		JNC	FWD	Is Carry exists? No, go to the label FWD
	0D				
	80				
800C	0C		INR	C	Make carry =1
800D	32	FWD	STA	8502 H	The contents of accumulator are stored in memory location 8502.
800E	02				
800F	85				
8010	79		MOV	A, C	Carry is moved to accumulator
8011	32		STA	8503 H	A Carry is stored in the location 8503
8012	03				
8013	85				
8014	76		HLT		Stop the execution

Ex: Input: **Ex :** 8500 – 67 D
8501 – 85 D

RESULT: 8502 – 52 D
8503 – 01 (Carry)

Example 4: Addition of two 16-bit numbers whose sum is 16 bits or more

Explanation: First 16-bit number is stored in two consecutive locations (Ex 8500 &8501) because in each location we can store only one 8-bit number. Store the second 16-bit number in the next two consecutive locations (For Ex: 8502 &8503).Add the LSB of the first number to the LSB of the second number and the MSB of the first number to the MSB of the second number using the DAD instruction. Store the sum in the next two locations and the carry (if any) in the third location

ALGORITHM:

Step1: First 16 bit number is in locations 8500 & 8501 respectively

Step2: Second 16-bit number is in locations 8502 & 8503

Step3: Add the two 16-bit numbers using DAD Instruction.

Step4: Sum is stored in locations 8504 & 8505.

Step5: Carry (if any) is stored in the location 8506.

Step6: Halt

PROGRAM:

ADDRESS	HEX – CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERAND	
8000	2A,00,85		LHLD	8500 H	First 16-bit number in H-L pair
8001	00				
8002	85				
8003	EB		XCHG		Exchange first number to D-E Pair
8004	2A		LHLD	8502 H	
8005	02				
8006	85				
8007	0E		MVI	00	MSB of the sum is initially 00
8008	00				
8009	19		DAD	D	Add two 16 –bit numbers
800A	D2		JNC	FWD	Is Carry? If yes go to the next line .Else go to the 800E LOCATION
800B	0E				
800C	80				
800D	OC		INR	C	Increment carry
800E	22	FWD	SHLD	8504 H	Store the LSB of the Sum in 8504 & MSB in 8505 locations
800F	04				
8010	85				
8011	79		MOV	A,C	MSBs of the sum is in

					Accumulator
8012	32		STA	8506 H	Store the MSB (Carry) of the result in 8506 location
8013	06				
8014	85				
8015	76		HLT		Stop execution

Ex: INPUT: 8500- 12 H LSB of the Ist Number

8501- 13 H MSB of the Ist Number

8502 -13 H LSB of the IInd Number

8503 -12H MSB of the IInd number

RESULT : 8504 - 25H LSB of the Sum

8505 – 25H MSB of the Sum

8506 -- 00 Carry .

Subtraction Programs:

Example 5: Subtraction of two 8-bit numbers without borrows.

Explanation: It's a simple program similar to addition of two 8- bit numbers, except that we use the instruction SUB instead of ADD. The first 8-bit number is stored in XX00 memory location and the second 8-bit number is stored in the XX01 location .Use the SUB instruction and store the result in the XX02 location.

ALGORITHM:

Step1. : Initialise H-L pair with the address of minuend.

Step2. : Move the minuend into accumulator

Step3. : Increment H-L pair

Step4. : Subtract the subtrahend in memory location M from the minuend.

Step5. : Store the result in XX02.

Step6. : Stop the execution

ADDRESS	HEX CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERAND	

8000	21		LXI	H, 8500	Initialise H-L pair and get the First number in to 8500 location
8001	00				
8002	85				
8003	7E		MOV	A,M	[A] ← [M]
8004	23		INX	H	[M+1] ← [M]
8005	96		SUB	M	A ← [A] – [M]
8006	23		INX	H	Next memory location
8007	77		MOV	M,A	Store the result in the location 8502
8008	76		HLT		Stop the execution

PROGRAM:

INPUT: Ex : 8500- 59H **Result: 8502 – 29H**

8501- 30H

Example 6: Subtraction of two 8-bit Decimal numbers.

Explanation: In this program we can't use the DAA instruction after SUB or SBB instruction because it is decimal adjust after addition only. So, for decimal subtraction the number which is to be subtracted is converted to 10's complement and then DAA is applied.

ALGORITHM:

Step1. : Initialise H-L pair with the address of second number (XX01).

Step2. : Find its ten's complement

Step3. : Decrement the H-L pair for the first number (XX00)

Step4. : Add the first number to the 10's complement of second number.

Step5. : Store the result in XX02.

Step6. : Stop the execution

PROGRAM:

ADDRESS	HEX CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERAND	
8000	21		LXI	H,8500	Initialise H-L pair and get theSecond number in to 8501 location
8001	00				
8002	85				
8003	3E		MVI	A,99	[A] ← 99
8004	99				
8005	96		SUB	M	9's complement of second number
8006	3C		INR	A	10's complement of second number
8007	2B		DCX	H	Address of the first number
8008	86		ADD	M	Add first number to 10's complement of second number
8009	27		DAA		
800A	32		STA	8502	Store the result in the location 8502
800B	02				
800C	85				
800D	76		HLT		Stop the execution

Ex: Input: 8500 -76 D **Result: 8502 - 41 D**

8501- 35 D

Example 6: Subtraction of two 16 –bit numbers.

Explanation: It is very similar to the addition of two 16-bit numers.Here we use SUB &SBB instructions to get the result .The first 16-bit number is stored in two consecutive locations and the second 16-bit number is stored in the next two consecutive locations.The lsbs are subtracted using SUB instruction and the MSBs aare subtracted using SBB instruction.The result is stored in different locations.

ALGORITHM:

Step1. : Store the first number in the locations 8500 & 8501.

Step2. : Store the second number in the locations 8502 &8503.

Step4. : Subtract the second number from the first number with borrow.

Step5. : Store the result in locations 8504 & 8505.

Step6. : Store the borrow in location 8506

Step 7: Stop the execution

PROGRAM:

Ex: INPUT : 8500- FF H LSB of the Ist Number RESULT: 8504 - 11H LSB

ADDRESS	HEX CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERAND	
8000	2A, 00,85		LHLD	8500 H	First 16-bit number in H-L pair
8003	EB		XCHG		Exchange first number to D-E Pair
8004	2A		LHLD	8502 H	Get the second 16-bit number in H-L pair
8005	02				
8006	85				
8007	7B		MOV	A, E	Get the lower byte of the First number in to Accumulator
8008	95		SUB	L	Subtract the lower byte of the second number
8009	6F		MOV	L, A	Store the result in L- register

800A			MOV	A, D	Get higher byte of the first number
800A	9C		SBB	H	Subtract higher byte of second number with borrow
800B	67		MOV	H, A	
800C	22		SHLD	8504	Store the result in memory locations with LSB in 8504 & MSB in 8505
800D	04				
80OE	85				
80OF	76		HLT		Stop execution

8501 - FF H MSB of the Ist Number

8502 -EE H LSB of the IInd Number

8503 –EE H MSB of the IInd number

8505 – 11 H MSB

Multiplication Programs

Example 7: Multiplication of two 8-bit numbers. Product is 16-bits.

Explanation: The multiplication of two binary numbers is done by successive addition. When multiplicand is multiplied by 1 the product is equal to the multiplicand, but when it is multiplied by zero, the product is zero. So, each bit of the multiplier is taken one by one and checked whether it is 1 or 0 .If the bit of the multiplier is 1 the multiplicand is added to the product and the product is shifted to left by one bit. If the bit of the multiplier is 0 , the product is simply shifted left by one bit. This process is done for all the 8-bits of the multiplier.

ALGORITHM:

Step 1 : Initialise H-L pair with the address of multiplicand.(say 8500)

Step 2 : Exchange the H-L pair by D-E pair. so that multiplicand is in D-E pair.

Step 3 : Load the multiplier in Accumulator.

Step 4 : Shift the multiplier left by one bit.

Step 5 : If there is carry add multiplicand to product.

Step 6 : Decrement the count.

Step 7 : If count ≠ 0; Go to step 4

Step 8 : Store the product i.e. result in memory location.

Step 9 : Stop the execution

PROGRAM:

ADDRESS	HEX - CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERAND	
8000	2A, 00,85		LHLD	H, 8500	Load the multiplicand in to H-L pair
8003	EB		XCHG		Exchange the multiplicand in to D-E pair
8004	3A		LDA	8502	Multiplier in Accumulator
8005	02				
8006	85				
8007	21		LXI	H.0000	Initial value in H-L pair is 00
8008	00				
8009	00				
800A	0E		MVI	C,08	Count =08
800B	08				
800C	29	LOOP	DAD	H	Shift the partial product left by one bit.
800D	17		RAL		Rotate multiplier left by one bit
800E	D2		JNC	FWD	Is Multiplier bit =1? No go to label FWD
800F	12				
8010	80				
8011	19		DAD	D	Product =Product +Multiplicand

8012	0D	FWD	DCR	C	COUNT=COUNT-1
8013	C2		JNZ	LOOP	
8014	0C				
8015	80				
8016	22		SHLD	8503	Store the result in the locations 8503 & 8504
8017	03				
8018	85				
8019	76		HLT		Stop the execution

INPUT :

Address	Data
8500	8AH – LSB of Multiplicand
8501	00 H – MSB of Multiplicand
8502	52 H - Multiplier

Result:	8503	34 H – LSB of Product
	8504	2C H – MSB of Product

Division Programs

Example 7: Division of a 16- bit number by a 8-bit number.

Explanation: The division of a 16/8-bit number by a 8-bit number follows the successive subtraction method. The divisor is subtracted from the MSBs of the dividend .If a borrow occurs, the bit of the quotient is set to 1 else 0.For correct subtraction process the dividend is

shifted left by one bit before each subtraction. The dividend and quotient are in a pair of register H-L.The vacancy arised due to shifting is occupied by the quotient .In the present example the dividend is a 16-bit number and the divisor is a 8-bit number. The dividend is in locations 8500 &8501.Similarly the divisor is in the location 8502.The quotient is stored at 8503 and the remainder is stored at 8504 locations.

ALGORTHM:

STEP1. : Initialise H-L pair with address of dividend.

STEP2. : Get the divisor from 8502 to register A & then to Reg.B

STEP3. : Make count C=08

STEP4. : Shift dividend and divisor left by one bit

STEP 5: Subtract divisor from dividend.

STEP6. : If carry = 1 : goto step 8 else step7.

STEP7. : Increment quotient register.

STEP8. : Decrement count in C

STEP9. : If count not equal to zero go to step 4

STEP10: Store the quotient in 8503

STEP11: Store the remainder in 8504

STEP12: Stop execution.

PROGRAM:

ADDR-ESS	HEX – CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERAND	
8000	21		LHLD	H, 8500	Initialize the H-L pair for dividend
8001	00				
8002	85				
8003	3A		LDA	8502 H	Load the divisor from location 8502 to accumulator
8004	02				
8005	85				
8006	47		MOV	B,A	Move Divisor to Reg.B from A
8007	0E		MVI	C,08	Count =08
8008	08				
8009	29	BACK	DAD	H	Shift dividend and quotient left by one bit

800A	7C		MOV	A,H	MSB of dividend in to accumulator
800B	90		SUB	B	Subtract divisor from MSB bits of divisor
800C	DA		JC	FWD	Is MSB part of dividend > divisor ? No,goto label FWD
800D	11				
800E	80				
800F	67		MOV	H,A	MSB of the dividend in Reg.H
8010	2C		INR	L	Increment quotient
8011	0D	FWD	DCR	C	Decrement count
8012	C2		JNZ	BACK	If count is not zero jump to8009 location
8013	09				
8014	80				
8015	22		SHLD	8503H	Store quotient in 8503 and remainder in 8504 locations
8016	03				
8017	85				
8018	76		HLT		Stop execution

Ex: **Input & Result**

Address	Data
8500	64 → LSB of Dividend
8501	00 → MSB of Dividend
8502	07 → Divisor
8503	0E → Quotient
8504	02 → Remainder

Largest & Smallest numbers in an Array

Example 8: To find the largest number in a data array

Explanation: To find the largest number in a data array of N numbers (say)first the count is placed in memory location (8500H) and the data are stored in consecutive locations.(8501….onwards).The first number is copied to Accumulator and it is compared with the second number in the memory location. The larger of the two is stored in Accumulator. Now the third number in the memory location is again compared with the accumulator. And the largest number is kept in the accumulator. Using the count, this process is completed , until all the numbers are compared .Finally the accumulator stores the smallest number and this number is stored in the memory location.85XX.

ALGORTHM:

Step1: Store the count in the Memory location pointed by H-L register.

Step2: Move the I st number of the data array in to accumulator

Step3: Compare this with the second number in Memory location.

Step4: The larger in the two is placed in Accumulator

Step5: The number in Accumulator is compared with the next number in memory .

Step 6: The larger number is stored in Accumulator.

Step 7; The process is repeated until the count is zero.

Step 8: Final result is stored in memory location.

Step 9: Stop the execution

PROGRAM

ADDR-ESS	HEX – CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERAND	
8000	21,00,85		LXI	H, 8500	INITIALISE H-L PAIR
8003	7E		MOV	C,M	Count in the C register
8004	23		INX	H	First number in H-L pair
8005	4E		MOV	A,M	Move first number in to Accumulator
8006	0D		DCR	C	Decrement the count
8007	91	LOOP1	INX	H	Get the next number
8008	BE		CMP	M	Compare the next number with previous number

8009	D2		JNC	LOOP2	Is next number >previous maximum?No,go to the loop2
800A	0D				
800B	80				
800C	7E		MOV	A,M	If,yes move the large number in to Accumulator
800D	0D	LOOP2	DCR	C	Decrement the count
800E	C2		JNZ	LOOP1	If count not equal to zero,repeat
800F	07				
8011	80				
8012	78				
8013	32		STA	85XX	Store the largest number in the location 85XX
8014	XX				
8015	85				
8016	76		HLT		Stop the execution

Ex : Input : 8500- N(Say N=7) **Result : 8508 - 7F**

8501-05

8502-0A

8503-08

8504-14

8505 -7F

8506-25

8507-2D

Example 9 : To find the smallest number in a data array.

Explanation: To find the smallest number in a data array of N numbers (say)first the count is placed in memory location (8500H) and the data are stored in consecutive locations.(8501….onwards).The first number is copied to Accumulator and it is compared with the second number in the memory location.The smaller of the two is stored in Accumulator.Now the third number in the memory location is again compared with the accumulator.and the smallest number is kept in the accumulator.Using the count,this process is completed until all the numbers are compared .Finally the accumulator stores the smallest number and this number is stored in the memory location.85XX.

ALGORTHM :

Step1: Store the count in the Memory location pointed by H-L register.

Step2: Move the I st number of the data array in to accumulator

Step3: Compare this with the second number in Memory location.

Step4: The smaller in the two is placed in Accumulator

Step5: The number in Accumulator is compared with the next number in memory .

Step 6: The smaller number is stored in Accumulator.

Step 7; The process is repeated until the count is zero.

Step 8: Final result is stored in memory location.

Step 9: Stop the execution

PROGRAM

ADDR-ESS	HEX – CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERAND	
8000	21		LXI	H, 8500	Initialise the H-L pair.
8001	00				
8002	85				
8003	7E		MOV	C,M	Count in the C register
8004	23		INX	H	First number in H-L pair
8005	4E		MOV	A,M	Move first number in to Accumulator
8006	0D		DCR	C	Decrement the count
8007	91	LOOP1	INX	H	Get the next number
8008	BE		CMP	M	Compare the next number with previous number
8009	D2		JC	LOOP2	Is next number <previous smallest ?If yes go to the loop2

800A	0D				
800B	80				
800C	7E		MOV	A,M	No,move the smaller number in to Accumulator
800D	0D	LOOP2	DCR	C	Decrement the count
800E	C2		JNZ	LOOP1	If count not equal to zero,repeat
800F	07				
8011	80				
8012	78				
8013	32		STA	85XX	Store the smallest number in the location 85XX
8014	XX				
8015	85				
8016	76		HLT		Stop the execution

Ex: Input : 8500 - N((Say N=7) **Result : 8508 – 04**

8501-09

8502-0A

8503-08

8504-14

8505 -7F

8506-04

8507-2D