8051 Micro Controller Block Diagram

The 8051 micro controller is an 8 bit micro controller. Basic components present inside 8051 microcontroller are

CPU (Central Processing Unit) :-

CPU act as a mind of any processing machine. It synchronizes and manages all processes that are carried out in micro controller. User has no power to control the functioning of CPU. It interprets the program stored in Rom and carries out from storage and then performs its projected duty. CPU manages the different types of registers available in 8051 micro controller.

Interrupts :-

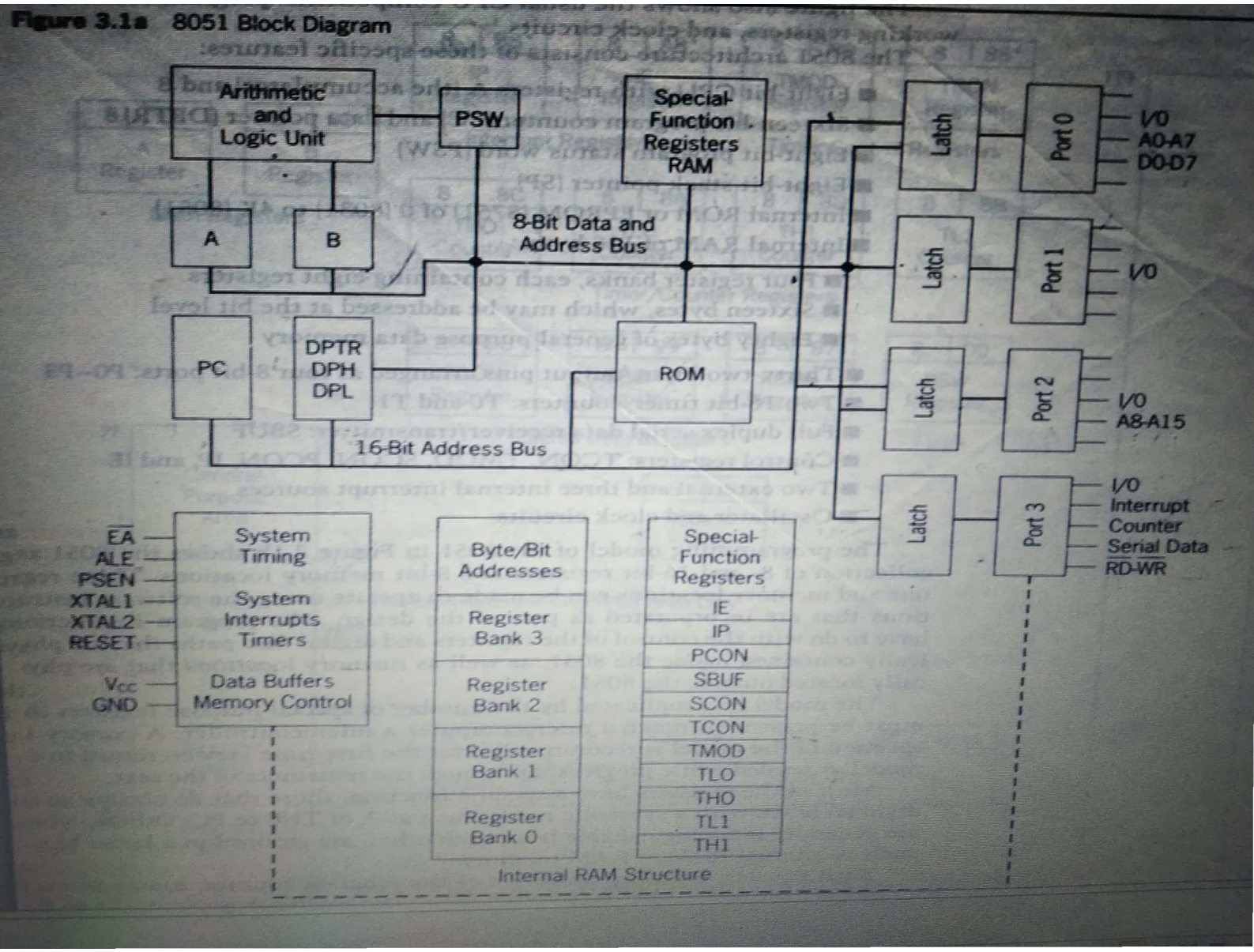
Interrupts is a sub-routine call that given by microcontroller when some other program with high priority is request for acquiring the system buses the n interrupts occur in current running program.

Interrupts provide a method to postpone or delay the current process, performs a Subroutine class/ task and then restart the standard program again.

Types of Interrupts in 8051 micro Controller :-

There are five types or sources of interrupts in

Figure 3.1a 8051 Block Diagram



8051 micro controller

1) Timer 0 overflow Interrupt - TF0
2) Timer 1 overflow Interrupt - TF1
3) External hardware Interrupt - INT0
4) External hardware Interrupt - INT1
5) Serial communication Interrupt - RI/TI

Memory :-

For operation micro controller requires a program. The program guides the micro controller to perform the specific tasks. This program installed in micro controller required some on chip memory for the storage of the program.

Micro controller also required memory for storage of data and operands for the short duration. In micro controller 8051 there is code or program memory of 4KB that is it has 4KB ROM and it also compromise of data memory (RAM) of 128 bytes.

Bus :-

Bus is a group of wires which uses a communication channel or acts as a means of data transfer. The different bus configuration includes 8, 16 or more

cables therefore a bus can bear 8 bits, 16 bits all together.

Types of busses in 8051 Micro controller:-

The two types of busses in 8051 controller are

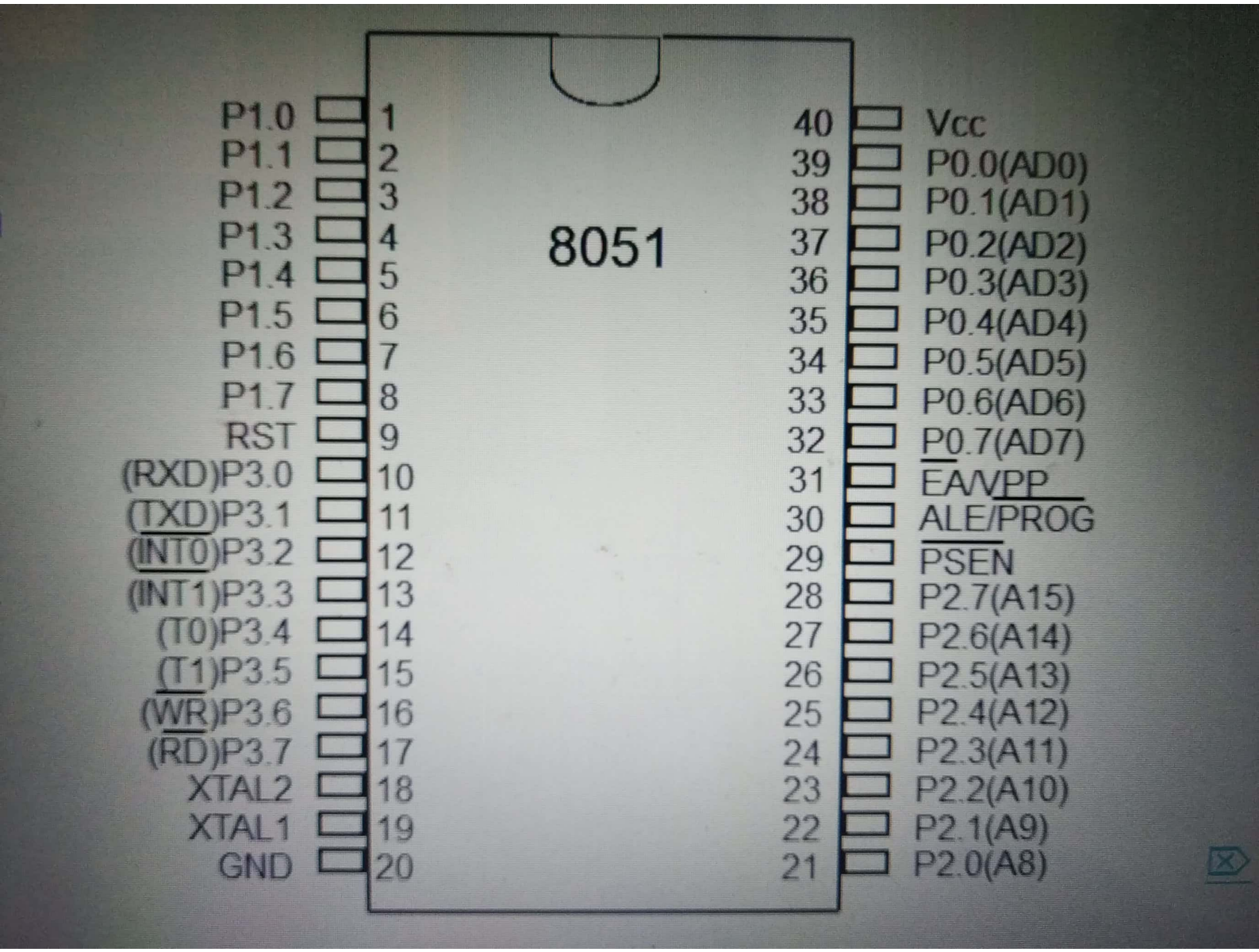
1) Address bus
2) Data bus.

Address bus:-

8051 micro controller is consisting of 16 bit address bus. It is generally used for transferring the data from cpu to memory.

DATA bus:-

8051 micro controller is consisting of 8 bit data bus. It is generally used for transferring the data from one peripherals position to other peripherals.



8051 Micro Controller - PIN DIAGRAM

Pins 1 to 8 :- These pins are known as port 1. This port does nt serve any functions It is internally pulled up. bi-directional I/o port.

Pin 9 :- It is a RESET pin, which is used to reset the micro controller to its initial values.

Pins 10 to 17 :- These pins are known as port 3. This port serve some functions like interupts, timer input, contol signals, serial communication signals RxD, TxD etc.

Pins 18 & 19 :- These pins are used for interfacing an external crystal to get system clock.

Pin 20 :- power supply to circuit.

Pins 21 to 28 :- These pins are port 2. It serves as I/O port. Higher order address bus signals are multiplexed.

Pin 29 :- This is program store enable pin. It is used to read a signal from external program memory.

Pin 30 :- This is external Access input. It is used to enable/disable the external memory interfacing.

Pin 31 :- This is Address latch enable pin- It is used to demultiplex the address-data signal of port.

Pins 32 to 39 :- These pins are known as port 0. It serves as I/o port. lower order address and data bus signals are multiplexed using this port.

Pin 40 :- This pin is used to provide power supply to the circuit

8051 Micro Controller - Addressing modes.

Addressing mode:-

The way in which the instruction is specified. In 8051 there are six types of addressing modes.

1) Immediate Addressing mode
2) Register Addressing mode
3) Direct Addressing mode
4) Register Indirect Addressing mode
5) Indexed Addressing mode
6) Implied Addressing mode.

1) Immediate Addressing mode:-

Immediate data is specified in the instruction itself.

```
MOV  A,#65H
MOV  DPTR,#2343H
MOV  R6,#65H
MOV  A,#A
MOV  P1,#65H.
```

2) Register Addressing mode:-

Either source or destination data must be present in a Register (R_0 to R_7).

```
MOV  Rn, A      n=0, ---7
ADD  A, Rn
MOV  ODL, R6
```

3) Direct Addressing mode:-

The source or destination address is specified by using 8-bit data in the instruction. Only the internal data memory can be used in this mode.

```
MOV  R0, 40H
MOV  56H, A.
MOV  A, 4.      ; ≡ MOV A, R4
MOV  6, 2.      ; Copy R2 to R6
```

MOV R6, R2 is invalid.

4) Register Indirect Addressing mode:-

In this mode, register is used as a pointer to the data.

```
MOV  A, @Ri     i=0 or 1
MOV  @R1, B
MOV  @R1, 80H
```

5) Indexed Addressing mode:-

In this mode the source memory can be accessed from program memory only. The destination operand is always register A.

MOVC A, @A+DPTR
MOVC A, @A+PC

The C in MOVC indicates code byte.

6) Implied Addressing mode:-

In this mode there will be a single operand. These type of instructions can work on specific registers only. These type of instructions are also known as register specific Instruction.

RLA (rotate register A content to left)
SWAP A (swap the nibbles in A).

8051 Instruction Set

There are 49 instruction Mnemonics in the 8051 micro controller and these are divided into five groups.

1) Data transfer
2) Arithmetic
3) Logical
4) boolean
5) program branching.

1) Data transfer Instructions:-

MOV
MOVC
MOVX
PUSH
POP
XCH
XCHD

2) Arithmetic Instructions:-

ADD - Addition without carry
ADC - Addition with carry

SUBB - Subtract with borrow.
INC - Increment by 1
DEC - decrement by 1
MUL - multiply
DIV - divide
DAA - Decimal Adjust Accumulator (A Register)

3) Logical Instructions:-
ANL - Logical AND
ORL - Logical OR
XRL - EX-OR
CLR - clear register
CPL - Complement the Register
RL - Rotate left
RLC - Rotate a byte and carry bit to left.
RR - Rotate right
RRC - Rotate a byte and carry bit to right
SWAP - Exchange lower and higher nibbles in byte.

4) Boolean or Bit manipulation Instructions:-
CLR - clear a bit (Reset to 0)
SETB - Set a bit (set to 1)

MOV – move a bit

JC - Jump if carry flag (Cf) is set

JNC – Jump if Cf is not set

JB - Jump if specified bit is set

JNB - Jump if specified bit is not set

JBC – Jump if specified bit is set and clear the bit.

ANL - bit wise AND.

ORL - bit wise OR.

CPL - Complement the bit.

5) Program branching Instructions:-

LJMP – long Jump (unconditional)

AJMP – Absolute Jump (un conditional)

SJMP – short Jump (unconditional)

JZ - Jump if A is equal to zero.

JNZ - Jump if A is not equal to zero.

CJNE – Compare and Jump if not equal.

NOP – NO operation

Lcall – long call to Subroutine

Acall – Absolute call to Subroutine.

RET - Return from Subroutine.

RETI - Return from interrupt

JMP - Jump to an Address (unconditional)

output:

I: 90h

I: 0X4D: 08 02 0F 00 01 02

V.R. SIDDHARTHA ENGINEERING COLLEGE

Programme Name : Arthmetic operations using Immediate Addressing mode

Branch : CSE

Date :

Sheet No. :

HEXADECIMAL					SYMBOLIC ASSEMBLER INSTRUCTIONS			
ADDRESS	1	2	3	4	LABEL	MNEMONIC	OPERAND	COMMENTS
						ORG	00h	
C:0X0000	74	05				MOV	A,#5h	5 is moved to A
C:0X0002	75	F0	03			MOV	B,#3h	3 is moved to B
C:0X0005	25	F0				ADD	A,B	B's content added with A's content
C:0X0007	F5	40				MOV	40h,A	A's content moved to Address 40h
C:0X0009	74	25				MOV	A,#5h	5 is moved to A
C:0X000B	95	F0				SUBB	A,B	Subtract B's content from A's
C:0X000D	F5	41				MOV	41h,A	A's content moved to 41h
C:0X000F	74	05				MOV	A,#5h	5 is moved to A
C:0X0011	A4					MUL	A,B	Multiply A's content with B's conten
C:0X0012	F5	42				MOV	42h,A	A's content moved to 42h
C:0X0014	85	F0	43			MOV	43h,B	B's content moved to 43h
C:0X0017	74	05				MOV	A,#5h	5 is moved to A
C:0X0019	75	F0	03			MOV	B,#3h	3 is moved to B
C:0X001C	84					DIV	A,B	Divide the contents of A with B
C:0X001D	F5	44				MOV	44h,A	A's content moved to 44h
C:0X001F	85	F0	45			MOV	45h,B	B's content moved to 45h
						END		

output:-

I: 60h

I: 0X60: 08 02 0F 00 01 02

V.R. SIDDHARTHA ENGINEERING COLLEGE

Branch : CSE

Programme Name : Arithmetic operations using Register Addressing mode

Date :

Sheet No. :

HEXADECIMAL					SYMBOLIC ASSEMBLER INSTRUCTIONS			
ADDRESS	1	2	3	4	LABEL	MNEMONIC	OPERAND	COMMENTS
						ORG	00h	
C:0X0000	79	05				MOV	R1,#5h	5 is moved to Register R1
C:0X0002	7A	03				MOV	R2,#3h	3 is moved to Register R2
C:0X0004	E9					MOV	A,R1	R1 contents moved to A
C:0X0005	2A					ADD	A,R2	R2 contents Added with A's contents
C:0X0006	F5	60				MOV	60h,A	A's content moved to 60h
C:0X0008	E9					MOV	A,R1	R1's content moved to A
C:0X0009	9A					SUBB	A,R2	contents of R2 subtracted from A
C:0X000A	F5	61				MOV	61h,A	A's content moved to 61h
C:0X000C	E9					MOV	A,R1	R1's content moved to A
C:0X000D	8A	F0				MOV	B,R2	R2 content moved to B
C:0X000F	A4					MUL	A,B	multiply A's content with B's
C:0X0010	F5	62				MOV	62h,A	A's content moved to 62h
C:0X0012	85	F0	63			MOV	63h,B	B's content moved to 63h
C:0X0015	E9					MOV	A,R1	R1 content moved to A
C:0X0016	8A	F0				MOV	B,R2	R2 content moved to B
C:0X0018	84					DIV	A,B	divide the contents of A with B
C:0X0019	F5	64				MOV	64h,A	A's content moved to 64h
C:0X001A	85	F0	65			MOV	65h,B	B's content moved to 65h
						END		

output:-

I: 50h

I: 0x50: 08 02 0f 00 01 02

V.R. SIDDHARTHA ENGINEERING COLLEGE

Programme Name : Arithmetic operations using Indirect Addressing mode

Branch : CSE

Date :

Sheet No. :

HEXADECIMAL | SYMBOLIC ASSEMBLER INSTRUCTIONS

ADDRESS	1	2	3	4	LABEL	MNEMONIC	OPERAND	COMMENTS
						ORG	00h	
C:0X0000	75	96	05			MOV	@R1, #5h	5 is pointed by R1 Register
C:0X0003	75	97	03			MOV	@R0, #3h	3 is pointed by R0 Register
C:0X0006	E5	96				MOV	A, @R1	data pointed by R1 is moved to A
C:0X0008	25	95				ADD	A, @R0	data pointed by R0 is added to A
C:0X000A	F5	70				MOV	50h, A	A's content moved to 50h
C:0X000C	E5	96				MOV	A, @R1	Move data pointed by R1 to A.
C:0X000E	95	97				SUBB	A, @R0	Subtract data pointed by R0 to A
C:0X0010	F5	71				MOV	51h, A	A's content moved to 51h
C:0X0012	E5	96				MOV	A, @R1	data pointed by R1 is moved to A
C:0X0014	85	97	F0			MOV	B, @R0	data pointed by R0 is moved to B
C:0X0017	A4					MUL	A, B	multiply A's content with B's.
C:0X0018	F5	72				MOV	52h, A	A's content moved to 52h
C:0X001A	85	F0	73			MOV	53h, B	B's content moved to 53h
C:0X001D	E5	96				MOV	A, @R1	data pointed by R1 is moved to A
C:0X001F	85	97	F0			MOV	B, @R0	data pointed by R0 is moved to B
C:0X0022	84					DIV	A, B	division of A with B.
C:0X0023	F5	74				MOV	54h, A	data in A moved to 54h
C:0X0025	85	F0	75			MOV	55h, B	data in B moved to 55h
						END		

Output:-

DI: 70h

I: 0X70: 08 02 0f 00 01 02

V.R. SIDDHARTHA ENGINEERING COLLEGE

Programme Name : Arithmetic operations using direct Addressing mode

Branch : CSE

Date :

Sheet No. :

HEXADECIMAL					SYMBOLIC ASSEMBLER INSTRUCTIONS			
ADDRESS	1	2	3	4	LABEL	MNEMONIC	OPERAND	COMMENTS
						ORG	00h	
C:0X0000	75	96	05			MOV	150,#5h	5 is moved to Address 150h
C:0X0003	75	97	03			MOV	151,#3h	3 is moved to Address 151h
C:0X0006	E5	96				MOV	A,150	150h contents Added to A
C:0X0008	25	97				ADD	A,151	151h contents Added to A
C:0X000A	F5	70				MOV	70h,A	move A's content to 70h
C:0X000C	E5	96				MOV	A,150	MOVE 150h contents to A
C:0X000E	95	97				SUBB	A,151	MOVE 151h contents to A
C:0X0010	F5	71				MOV	71h,A	Move A's contents to 71h
C:0X0012	E5	96				MOV	A,150	MOVE 150h contents to A
C:0X0014	85	97	F0			MOV	B,151	MOVE 151h contents to B
C:0X0017	A4					MUL	A,B	Multiply A with B
C:0X0018	F5	72				MOV	72h,A	Move A contents to 72h
C:0X001A	85	97	F0			MOV	73h,B	MOVE B contents to 73h
C:0X001D	F5	96				MOV	A,150	MOVE 150h contents to A
C:0X001F	85	97	f0			MOV	B,151	move 151h contents to B
C:0X0022	84					DIV	A,B	Divide A with B
C:0X0023	F5	74				MOV	74h,A	MOVE A's content to 74h
C:0X0025	85	F0	75			MOV	75h,B	move B's content to 75h
						END		

output:-

I:30h

I: 0X30: 01 02 03 04 05 06 07 08 09 00

I: 40h

I: 0X40: 01 02 03 04 05 06 07 08 09 00

V.R. SIDDHARTHA ENGINEERING COLLEGE

Programme Name : Moving data from internal to internal memory

Branch : CSE

Date : .....

Sheet No. : .....

HEXADECIMAL					SYMBOLIC ASSEMBLER INSTRUCTIONS			
ADDRESS	1	2	3	4	LABEL	MNEMONIC	OPERAND	COMMENTS
						ORG	00h	
C:0X0000	78	30				MOV	R0,#30h	30 is moved to R0
C:0X0002	79	40				MOV	R1,#40h	40 is moved to R1
C:0X0004	7A	0A				MOV	R2,#0Ah	0A is moved to R2
C:0X0006	E6				back	MOV	A,@R0	move data pointed by R0 to A
C:0X0007	F7					MOV	@R1,a	move data pointed by R1 to A
C:0X0008	08					INC	R0	Increment the content of R0
C:0X0009	09					INC	R1	Increment the content of R1
C:0X000A	0A	FA				DJNZ	R2,back	
						END		

Output:-

I: 20H

I: 0X20: 01 02 03 04 05 06 07 08 09 00

X: 2000H

X: 0X002000: 01 02 03 04 05 06 07 08 09 00

V.R. SIDDHARTHA ENGINEERING COLLEGE

Branch : CSE

Programme Name : Moving data from internal to External memory

Date :

Sheet No. :

HEXADECIMAL					SYMBOLIC ASSEMBLER INSTRUCTIONS			
ADDRESS	1	2	3	4	LABEL	MNEMONIC	OPERAND	COMMENTS
						ORG	00h	
C:0x0000	74	00				MOV	A,#00h	00 is moved to A
C:0x0002	78	30				MOV	R_0,#30h	30 is moved to R_0
C:0x0004	90	20	00			MOV	dptr,#2000h	2000 is moved to dptr
C:0x0007	7A	0A				MOV	R_2,#0ah	0a is moved to R_2
C:0x0009	E6				back	MOV	A,@R_0	data pointed by R_0 is moved to A
C:0x000A	F0					MOVX	@dptr,A	A is pointed by DPTR
C:0x000B	08					INC	R_0	R_0 is incremented
C:0x000C	A3					INC	dptr	dptr is incremented
C:0x000D	DA	FA				DJNZ	R_2,back	
						END		

output:-

X: 2000h

X: 0X002000: 01 02 03 04 05

I: 30h

I: 0X30: 01 02 03 04 05

V.R. SIDDHARTHA ENGINEERING COLLEGE

Programme Name : Moving data from external to internal memory

Branch : CSE

Date :

Sheet No. :

HEXADECIMAL					SYMBOLIC ASSEMBLER INSTRUCTIONS			
ADDRESS	1	2	3	4	LABEL	MNEMONIC	OPERAND	COMMENTS
						ORG	00h	
C: 0X0000	78	30				MOV	R0, #30h	30 is moved to dpte
C: 0X0002	90	20	00			MOV	dpte, #2000h	2000 is moved to dpte
C: 0X0005	7A	05				MOV	R2, #05h	05 is moved to R2
C: 0X0007	E0				back	MOVX	A, @dpte	dpte is pointed to A
C: 0X0008	F6					MOV	@R0, A	a is pointed by R0
C: 0X0009	A3					INC	dpte	dpte is incremented
C: 0X000A	08					INC	R0	R0 is incremented
C: 0X000B	DA	FA				DJNZ	R2, back	
						END		

output:-

X: 4200H

X: 0X004200 : 01 02 03 04 05

X: 4300H

X: 0X004300 : 01 02 03 04 05

V.R. SIDDHARTHA ENGINEERING COLLEGE

Branch : C.S.E

Programme Name : ~~Add~~ Moving data from external memory to external memory

Date :

Sheet No. :

HEXADECIMAL | **SYMBOLIC ASSEMBLER INSTRUCTIONS**

ADDRESS	1	2	3	4	LABEL	MNEMONIC	OPERAND	COMMENTS
						ORG	00H	
C:0X0000	78	05				MOV	R0,#05H	move 05 to R0
C:0X0002	79	00				MOV	R1,#00H	move 00 to R1
C:0X0004	7A	00				MOV	R2,#00H	move 00 to R2
C:0X0006	75	83	42		UP	MOV	DPH,#42H	move 42H to DPH
C:0X0009	89	82				MOV	DPL,R1	move R1 content to DPL
C:0X000B	E0					MOVX	A,@DPTR	dptr is pointed to A
C:0X000C	75	83	43			MOV	DPH,#43H	43H moved to DPH
C:0X000F	8A	82				MOV	DPL,R2	R2 content moved to DPL
C:0X0011	F0					MOVX	@DPTR,A	A is pointed by DPTR
C:0X0012	09					INC	R1	Increment R1
C:0X0013	0	A				INC	R2	Increment R2
C:0X0014	D8	F0				DJNZ	R0,UP	Decrement R0 and Jump to UP if not zero
						END		

output:-

I: 50h

I: 0X50: 02 03 01 0A 82

output:-

X: 3500h

X: 0X003500: FD

V.R. SIDDHARTHA ENGINEERING COLLEGE

Programme Name : Logical & Rotate operations

Branch : CSE

Date : .....

Sheet No. : .....

HEXADECIMAL					SYMBOLIC ASSEMBLER INSTRUCTIONS			
ADDRESS	1	2	3	4	LABEL	MNEMONIC	OPERAND	COMMENTS
						ORG	00h	
C:0X0000	74	02				MOV	A,#02h	MOVE 02 to A
C:0X0002	75	F0	03			MOV	B,#03h	MOVE 03 to B
C:0X0005	55	F0				ANL	A,B	ANL A and B
C:0X0007	F5	50				MOV	50h,A	MOVE A'S content to 50h
C:0X0009	74	02				MOV	A,#02h	MOVE 02 to A
C:0X000B	45	F0				ORL	A,B	ORL A and B
C:0X000D	F5	51				MOV	51h,A	MOVE A's content to 51h
C:0X000F	74	02				MOV	A,#02h	MOVE 02 to A
C:0X0011	65	F0				XRL	A,B	XRL A and B
C:0X0013	F5	52				MOV	52h,A	MOVE A'S content to 52h
C:0X0015	74	05				MOV	A,#05h	MOVE 05 to A
C:0X0017	23					RL	A	RAtate left A
C:0X0018	F5	53				MOV	53h,A	MOVE A's content to 53h
C:0X001A	74	05				MOV	A,#05h	MOVE 05 to A
C:0X001C	03					RR	A	Rotate Right A
C:0X001D	F5	F4				MOV	54h,A	MOVE A'S content to 54h
						END		
					is complement			
						ORG	00h	
C:0X0000	74	02				MOV	A,#02h	MOVE 02h to A
C:0X0002	F4					CPL	A	Complement A
C:0X0003	90	35	00			MOV	dpte,#3500h	move 3500 to dpte
C:0X0006	F0					MOVX	@dpte,A	A is pointed by dpte
						END		

Output:-
X: 3000h
X: 0X003000: 78

output:-
X: 3500h
X: 0X003500: FE

output:-
X: 3500h
X: 0X003500 : 06

Date : .....

SHEET N-

V.R. SIDDHARTHA ENGINEERING COLLEGE

Programme Name : factorial of a Number

Branch : CSE

Date : .....

Sheet No. : .....

HEXADECIMAL					SYMBOLIC ASSEMBLER INSTRUCTIONS			
ADDRESS	1	2	3	4	LABEL	MNEMONIC	OPERAND	COMMENTS
C:0X0000	78	05				ORG MOV	00h R0,#05h	5 is moved to R0
C:0X0002	74	01				MOV	A,#01h	01 is moved to A.
C:0X0004	88	F0			BACK	MOV	R,R0	R0 contents moved to B
C:0X0006	A4					MUL	AB	multiply A,B contents
C:0X0007	D8	FB				DJNZ	R0,BACK	decrement R0 and jump if not zo
C:0X0009	90	30	00			MOV	dptr,#3000h	Move 3000 to Dptr
C:0X000C	F0					MOVX	@dptr,A	A is pointed by dptr.
						END		
					2's complement			
						ORG	00h	
C:0X0000						MOV	A,#02h	02 is moved to A
C:0X0002						CPL	A	complement A
C:0X0003						INC	A	Increment A.
C:0X0004						MOV	dptr,#3500h	move 3500 to dptr
C:0X0007						MOVX	@dptr,A	A is pointed by dptr
						END		
					Binary to Gray Conversion			
						ORG	00h	
C:0X0000						MOV	A,#04h	MOVE 04 h to A.
C:0X0002						MOV	R,A.	MOVE A content to R.
C:0X0004						RR	A	Rotate Right A.
C:0X0005						XRL	A,B	XRL A and B.
C:0X0007						MOV	dptr,#3500h	MOVE 3500 to dptr
C:0X000A						MOVX	@dptr,A	A is pointed by dptr
						END		

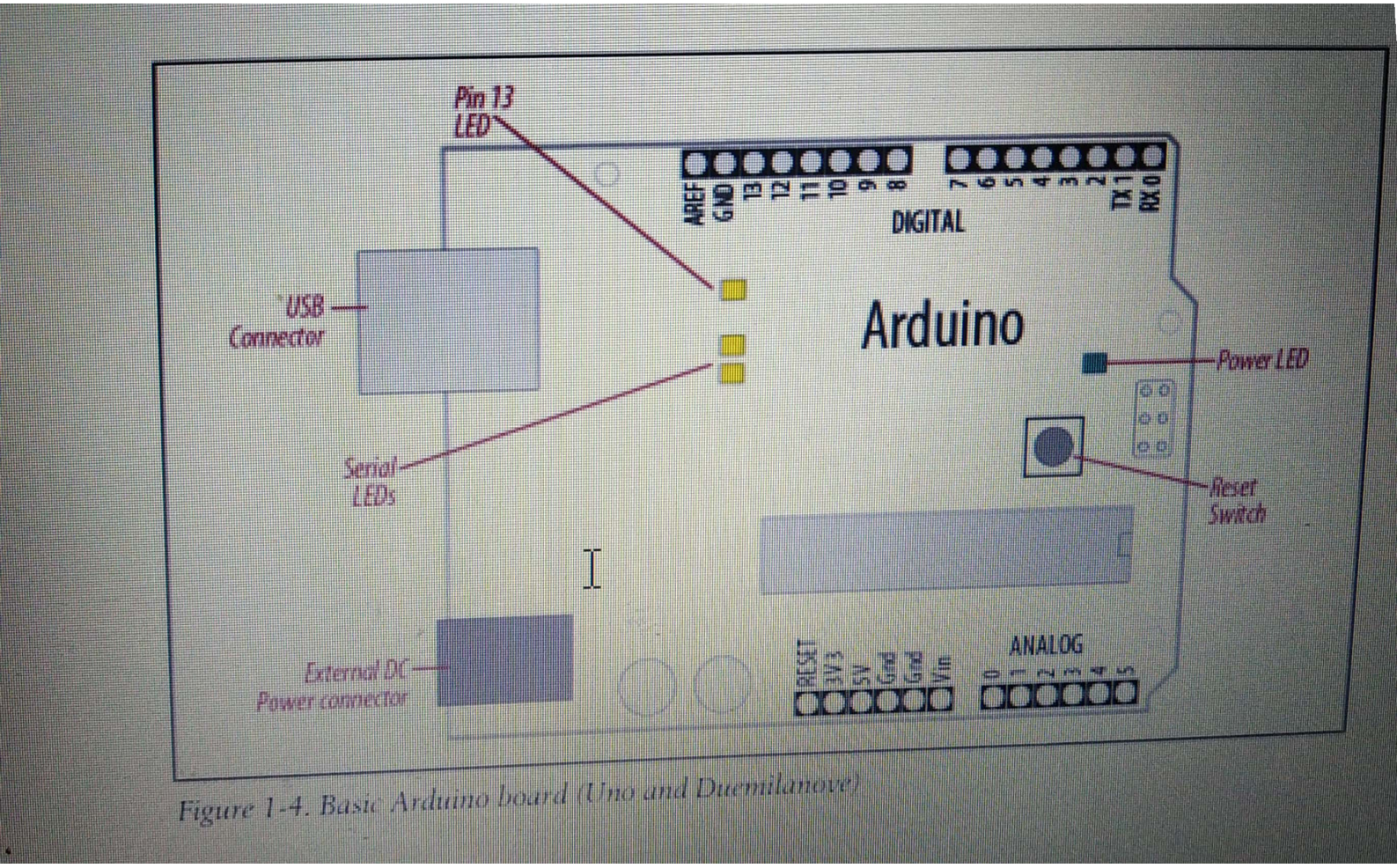


Figure 1-4. Basic Arduino board (Uno and Duemilanove)

Date : .....

ARDUINO UNO

Arduino UNO is a microcontroller based on 8 bit ATmega 328p microcontroller. Along with the ATmega328P it consists of other components such as crystal oscillator, Serial communication, voltage regulator, etc to support the microcontroller. Arduino UNO has 14 digital input/output pins (out of which 6 can be used as PWM outputs), 6 analog input pins, A USB connection, A power barrel Jack, an ICSP header and a reset button.

14 digital input/output pins can be used as input or output pins by using pinMode(), digitalRead() and digitalWrite() functions in arduino programming.

Out of 14 pins some pins have specific functions as listed below:

* Serial pins 0(Rx) and 1(Tx):

Rx and Tx pins are used to receive and transmit TTL Serial data. They are connected with the corresponding ATmega 328p USB to TTL Serial chip.

* External Interrupt Pins 2 and 3:-

These pins can be configured to trigger an interrupt on a low value, a rising or falling edge or a change in value.

* PWM pins 3, 5, 6, 9 and 11:-

These pins provide an 8-bit pwm output by using analogWrite() function.

* SPI pins 10(SS), 11(MOSI), 12(MISO) and 13(SCK):-

These pins are used for spi Communication.

* IN-built LED PIN 13:-

This pin is connected with the built in LED, when pin 13 is high - LED is on and when pin 13 is low it is off

Along with 14 digital pins there are 6 analog input pins, each of which provide 10 bits of resolution i.e 1024 different values. They measure from 0 to 5 volts but this limit can be increased by using AREF PIN with analogReference() function.

* Analog pin 4 (SDA) and pin 5 (SCA) also used for TWI communication using wire library.

Arduino UNO has a couple of other pins as explained below

* AREF :- used to provide reference voltage for analog inputs with analog Reference() function.
* Reset pin :- Making this pin low resets the micro-controller.

Blinking of LED

Aim:- To write a program to perform blinking of LED using Arduino.

Description:-

The light emitting diode is a p-n junction diode. It is specially doped diode and made up of a special type of semiconductors. when the light emits in the forward biased, then it is called as a light emitting diode.

Components required:-

Arduino uno
connecting wires
LED
Resistor

Program:-

```
void setup()
 {
  pinMode(8, OUTPUT);
 }
void loop()
 {
  digitalWrite(7, HIGH);
  delay(3000);
```

```
digitalWrite(7,LOW);
delay(1000);
}
```

RESult:- Arduino program to peeform blinking of LED is implemented Successfully.

Date : ..... SHEET No. .....

Intruder Detection using PIR Motion Sensor.

Aim:- To write a Arduino program to perform Intruder Detection using PIR motion Sensor.

DESCription:-

A PIR or a passive Infared Sensor can be used to detect the presence of human beings in its proximity. A PIR sensor detects the Infeared light radiated by a warm object. It consists of pyro electric sensors which introduce changes in their temperature into electric signal. when infeared light strikes a crystal, it generates an electric charge. Thus a PIR Sensor can be used to detect presence of human beings within a detection area of approximately 14 meters.

Components required:-

ARDUINO
PIR sensor
LED
Resistor, bread board
Connecting wires.

Program:-

```
int S=0;
void setup()
{
 PinMode(6,INPUT);
 pinMode.(7,OUTPUT);
 Serial.begin(9600);
}
void loop()
{
 S= digitalRead.(6);
 if(S==HIGH)
 {
  digitalWrite(7,HIGH);
  Serial.println("detected");
 }
 else
 {
 digitalWrite(7,LOW);
 Serial.println("not detected");
 }
delay(10);
}
```

Result:- Arduino program to perform Intruder detection using PIR Sensor is implemented Successfully.

Date : .....

SHEET No. .....

DISTANCE Measurement using UltraSonic Sensor

Aim:- To write a Arduino Program to preform distance measuring using ultra Sonic Sensor.

Description:-

An ultraSonic Sensor is a device that can measure the distance to an object by using Sound waves. It measures distance by Sending out a Sound wave at a specific frequency and listening for that Sound wave to bounce back. by recording the elapsed time between the Sound wave being generated and the Sound wave bouncing back, it is possible to calculate the distance b/w the Sonar Sensor and the object.

Components required:-

Arduino

ultrasonic distance Sensor

connecting wires.

Program:-

```
int cm =0
int inches=0;
long readUltrasonicDistance(int triggerPin, int echoPin)
{
```

```
digitalWrite(triggerpin, LOW);
delayMicroseconds(2);
digitalWrite(triggerpin, HIGH);
delayMicroseconds(10);
digitalWrite(echopin, INPUT);
return pulseIn(echopin, HIGH);
}
Void setup()
{
 Serial.begin(9600);
}
Void loop()
{
 cm = 0.01723 * readultrasonicDistance(3,3);
 inches = (cm/2.54);
 Serial.print(inches);
 Serial.print("in\n");
 Serial.print(cm);
 Serial.println("cm");
delay(1000);
 if (inches <1).{
  }
}
```

Date : .....

SHEET No. .....

Result:- Arduino program to peaform distance measurement using ultrasonic sensor is implemented Successfully.