Scoring Indicators

CODE:TED[15] 5041 VERSION (SETA)

(I)

1.ATmega32,ATmega16 2

2.32bytes , 64bytes 2

3.MOV Rd,Rr , LDI Rd,k LDS Rd,k (or any other) 2

4. int, char, unsigned char, signed char, short, long (any two) 2

5. automobile,avionics,aerospace,telecommunication (any two) 2

PART - B
(Maximum marks: 30)

II

1. AVR micro controllers have 32 general purpose 8-bit registers, named R0 to R31. All arithmetic operations operate on those registers, only load and store instructions access RAM. A limited number of instruction operate on 16-bit register pairs. The lower-numbered register of the pair holds the least significant bits and must be even-numbered. The last three register pairs are used as pointer registers for memory addressing. They are known as X (R27:R26), Y (R29:R28) and Z (R31:R30). Post increment and pre decrement addressing modes are supported on all three. Y and Z also support a six-bit positive displacement.

(6)

2. 32K bytes of In-System Programmable Flash Program memory with Read-While-Write Capabilities,1024 bytes EEPROM ,2K byte SRAM,32 general purpose I/O lines,JTAG interface for Boundary scan,On-chip Debugging support and programming,3 flexible Timer/Counters with compare modes,Internal and External Interrupts,Serial programmable USART,Byte oriented Two-wire Serial Interface (I2C),8-channel, 10-bit ADC,Programmable Watchdog Timer with Internal Oscillator,SPI serial port,6 software selectable power saving modes

(Any 6 poins) 6

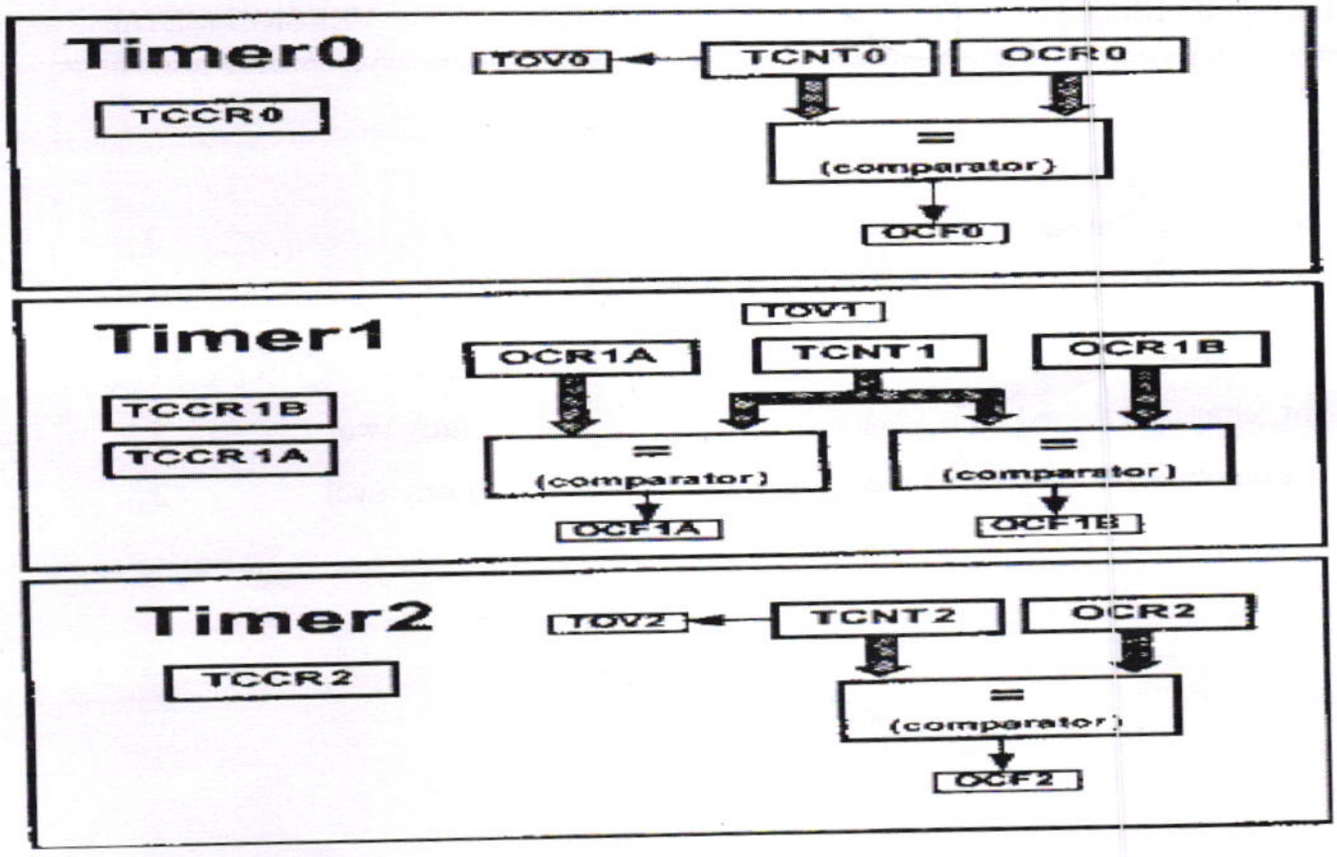
3. for addition(ADD Rd,Rr) /add with carry(ADC Rd,Rr)/substract(SUB Rd,Rr)/(substact with borrow)SBC Rd,Rr arrithmatic (any three/any other) Logical OR(OR Rd,Rr), Logical AND(AND Rd,Rr),Ex or(EOR Rd,Rr) logical

(any three arithmetic & logic format) 6

4. For the ATmega32 40-pin chip 32 Pins are available for I/O operation. The four ports PORTA, PORTB, PORTC, and PORTD are programmed for performing desired operation. Each port in AVR microcontroller has three I/O registers associated with it. They are designated as PORTx, DDRx and PINx. For example: - in case of Port B we have PORTB, DDRB, and PINB. Here DDR stands for Data Direction Registers, and PIN stands for Port Input pins.

(Explanations) 6

5.



(diagram not necessary required explanations only needed)

Atmega32 has 3 timer units, timer 0, timer 1 and timer 2 respectively Timer 0 is a 8 bit timer. It basically means it can count from 0 to 2^8 255. The operation of timer 0 is straight forward. The TCNT0 register hold the timer Count and it is incremented on every timer **"tick"**. If the timer is turned on it ticks from 0 to 255 and overflows. If it does so, a Timer OverFlow Flag(TOV) is set. The Timer 1 is 16 bit, that means it can count from 0 to $$2^{16} = 65536$$. Hence the Timer/Counter 1 is a 16 bit registered formed out of TCNT1H and TCNT1L

(6)

6. The USART stands for universal synchronous and asynchronous receiver and transmitter. It is a serial communication of two protocols. This protocol is used for transmitting and receiving the data bit by bit with respect to clock pulses on a single wire. The AVR microcontroller has two pins: TXD and RXD, which are specially used for transmitting and receiving the data serially. Any AVR microcontroller consists of USART protocol with its own features. The AVR microcontroller of USART protocol operates in three modes which are:Asynchronous Normal Mode, Asynchronous Double Speed Mode ,Synchronous Mode

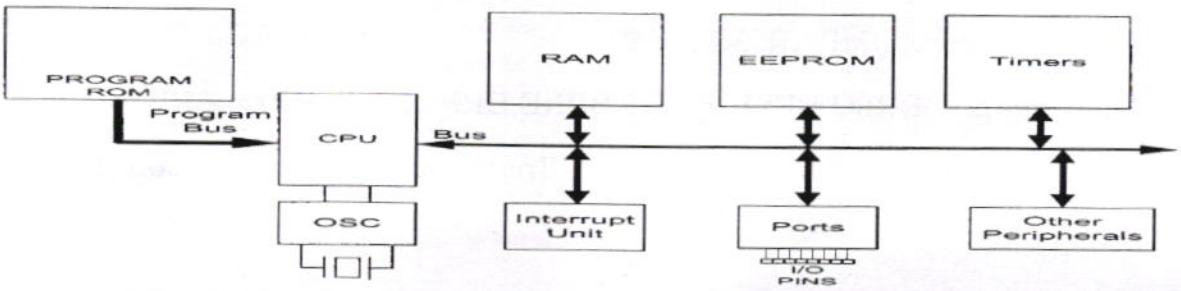
(Explanations) 6

7.Embedded NT –it only works with x86/pentium processor,require 9mb RAM 8mb ROM
Window XP Embedded,-it support IEEE 802.11 ,infrared interface,IPV6 ,successor to embedded NT.
Embedded Linux –open source software, POSIX support, large S/W resources available

(3 OS name-3,Explanation-3) 6

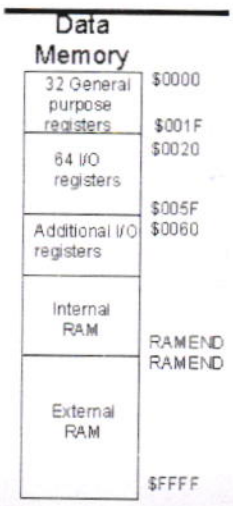
UNIT – I

III.a.



(Block diagram -6 Explanation-2) (8)

b.



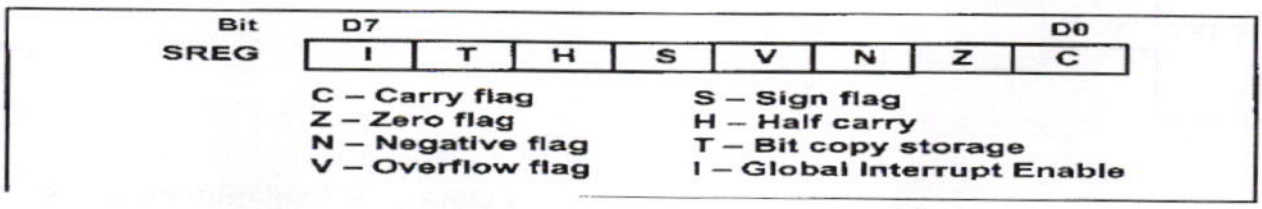
(Diagram -4 expalnation-3) (7)

IV.a

Register addressing mode,direct addressing mode,Register indirect addressing mode

(modes 4 explanation 4) (8)

b.



(Format 4 explain3) (7)

UNIT – II

V.a.

```
    CBI DDRA,0        ;SET PA0 AS INPUT PIN
    SBI DDRA,7        ;SET PA7 AS OUTPUT PIN
 XX  SBIC PINA,0        ;SKIP NEXT IF PB0=0
   RJMP XXX          ;JUMP TO XXX IF PB0=1
   CBI PORTA,7       ;LED OFF
   RJMP XX           ;JUMP TO XX IF PA0=0(SWITCH OFF)
XXX SBI PORTA,7         ;LED ON
   RJMP XX           ; JUMP TO XX IF PA0=1(SWITCH ON)
```

(use any other algorithm also)(8)

b. branching- conditional & unconditional

unconditional jump: JMP ,RJMP,IJMP

conditional jump: BRLO,BRSH,BREQ,BRNE,BRMI,BRPL,BRVS,BRVC

(Instructions-4 Explanation-3) (7)

OR

VI.a.

```
        CBI DDRB,3      ;set portb pin3 as input
        SBI DDRC,5      ;set portc pin 5 as output
LABEL2  SBIC PINB,0     ;skip rjmp if pin3 of portb=0(switch off)
        RJMP  LEBAL1    ;jump to LEBEL1 if pinb port3 is 1(switch on)
        CBI PORTC,5     ;portc pin5=0(buzzer off)
        RJMP LEBEL2
LABEL1  SBI PORTC,5     ;portc pin5=1(buzzer on)
        RJMP LABEL2
```

(Or any suitable program) (8)

b.

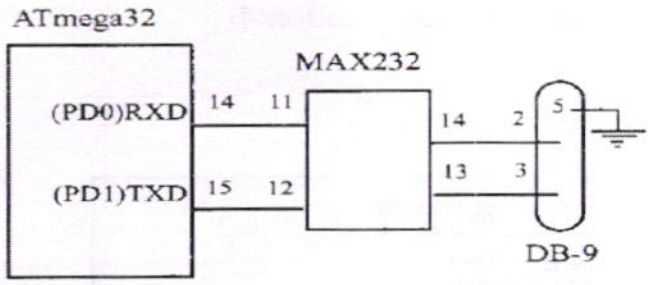
Macros allow the programmer to write the task once only ,and to invoke it whenever it is needed **subroutine** is a sequence of program instructions that perform a specific task, packaged as a unit. This unit can then be used in programs wherever that particular task should be performed.

```
.MACRO name
----------(Body of the macro)
.ENDMACRO
```

(Explanation-7) (7)

UNIT – III

VII.a.



(Diagram -4 Explanation-4) (8)

b.

In the *interrupt* method, whenever any device needs the microcontroller's service, the device notifies it by sending an interrupt signal. Upon receiving an interrupt signal, the microcontroller stops whatever it is doing and serves the device. The program associated with the interrupt is called the *interrupt service routine* (ISR)

If two interrupts are activated at the same time, the interrupt with the higher priority is served first. The priority of each interrupt is related to the address of that interrupt in the interrupt vector. The interrupt that has a lower address, has a Higher priority

(Explanations-7) (7)

VIII.a.

&	Bitwise AND operator
^	Bitwise XOR operator
|	Bitwise OR operator
&&	Logical And operator
||	Logical Or operator
?:	Conditional operator

(Operator 5 explanation 3) (8)

b.

```
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
 DDRB = 0xFF; //makes PORTB as Output
 while(1) //infinite loop
 {
  PORTB = 0xFF; //Turns ON  LEDs
  _delay_ms(2); //2 milli second delay
  PORTB= 0x00; //Turns OFF  LEDs
  _delay_ms(2); //2 milli second delay
 }
}
```

(Consider any other program also) (7)