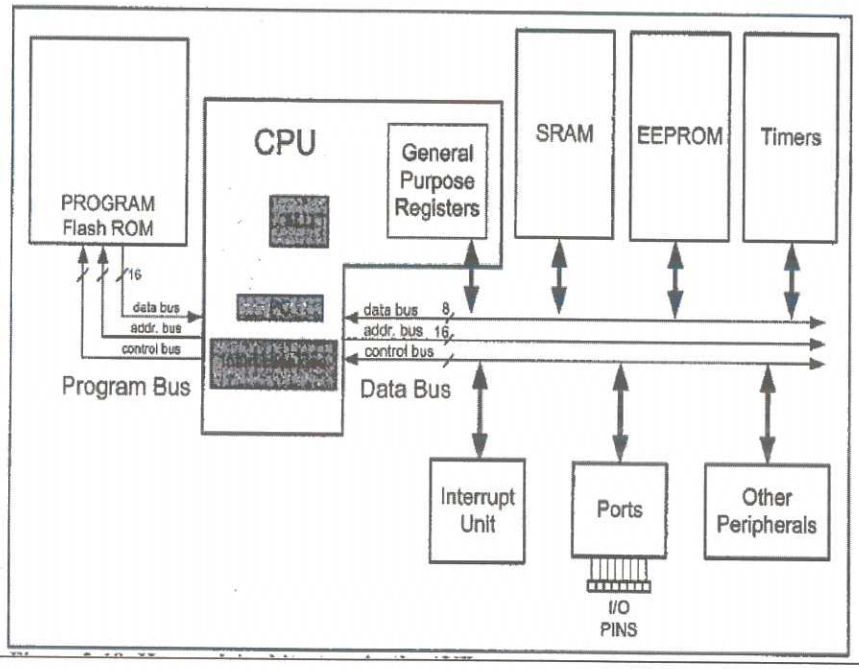
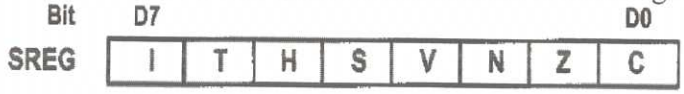
SCHEME OF VALUATION

Revision 2015 Course Title : MICRO CONTROLLERS				Course Code 6132
Q.No	Scoring Indicator	Split up score	Sub Total	Total
I	**PART A**			
1	Hex-0xFF,Decimal-255	2		
2	1. Increase the clock frequency of the chip. 2. Use Harvard architecture by increasing the number of buses to bring more information(code and data) into the CPU to be processed. 3. Change the internal architecture of the CPU and use what is called RISC architecture.	Any 2		
3	32	2		
4.	The group of memory locations set aside to hold the address f ISRs is called the Interrupt Vector Table (IVT).	2		10
5	UCSRs are 8-bit control registers used for controlling serial communication in the AVR.	2		
II	**PART B**			
1	JMP JMP instruction can go to any memory location in the 4M(word) address space of the AVR.It is a 4 byte (32 bit) instruction in which 10bits are used for the opcode, and the other 22 bit represent the 22bit address of the target location. RJMP It is 2byte instruction, the first 4bits are the opcode and the rest (lower 12 bits) is the relative address of the target location. The relative address range of 0x00-$FFF is divided into forward and backward jumps. If the jump is forward, then the relative address is positive. If the jump is backward, then the relative address if negative. IJMP IJMP is a 2 byte instruction. When the instruction executes, the PC is loaded with the contents of the Z register, so it jumps to the address pointed by the Z register.	2 2 2		6

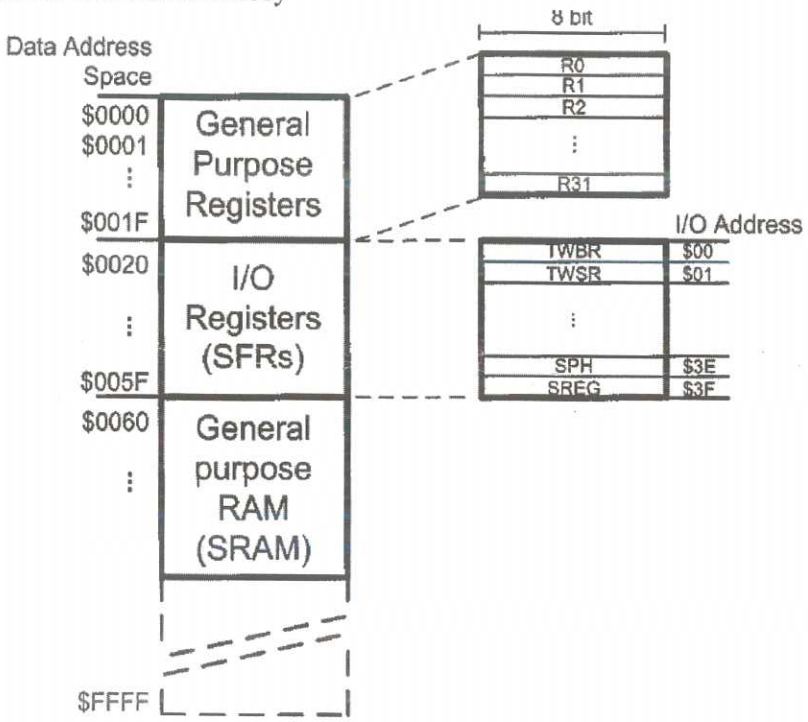
2	**Harvard Architecture** AVR uses Harvard architecture, which means that separate buses for the code and the data. The Program Bus provides access to the program flash Rom where as Data Bus is used for bringing data to the CPU. Program Bus Data bus is 16 bit s wide and the address bus is as wide as the PC register to enable the CPU to address the entire Program Flash ROM. Data Bus The data bus is 8 bits wide, The address bus is 16 bits wide.  	1 1 1 3		6
3	SBI DDRB,2 ; bit =1 ,make PB2 ad an output pin AGAIN: SBI PORTB,2 ; set bit (PB2= high) CALL DELAY CBI PORTB,2 ; clear bit (PB2=low) CALL DELAY RJMP AGAIN define delay function appropriately	1 1 1 1 2		6

4	1. Using simple for loop
	Ex: void delayms(void)
	{
	For(i=0;i<255;i++);
	}
	Function call : delayms();
	2. Using predefined function
	Ex: Call the predefined function directly _delay_ms(d)
	Header file is <util/delay.h>
	3. Using AVR timers
	
	explanation for each
		1
		1
		1
		3
				6

5	1. It finishes the instruction it is currently executing and saves the address of the next instruction(Program counter) 2. It jumps to a fixed location in memory called the interrupt vector talble. The interrupt vector table directs the microcontroller to the address of the interrupt service routine (ISR). 3. The microcontroller starts to execute the interrupt service routine until it reaches the last instruction of the subroutine, which is RETI(return from interrupt). 4. Upon executing RETI instruction, microcontroller return to the place where it is interrupted. First it get PC address from the by popping the top bytes to the stack into the PC. Then it starts to execute from that address.	1 2 1 2		6
6	Both perform the same action of popping off the top bytes of the stack into the program counter, and making the AVR return to where it left off. However RETI also performs the additional task of setting the I flag, indicating that the servicing of the interrupt is over and the AVR now can accept a new interrupt if use RET instead of RETI as the last instruction of the interrupt service routine, simply block any new interrupt after the first interrupt, because the I would indicate that the interrupt is still being serviced.	3 3		6
7	1. The UCSRB register is loaded with the value 08H, enabling the USART transmitter. The transmitter will override normal port operation for the TxD pin when enabled. 2. The UCSRC register is loaded with the value 06H, indicating asynchronous mode with 8-bit data frame, no parity, and one stop bit. 3. The UBRR is loaded with the value to set the baud rate for serial data transfer. 4. The character byte to be transmitted serially is written into the UDR register. 5. Monitor the UDRE bit of the UCSRA register to make sure UDR is ready for the next byte. 6. To transmit the next character, go to step 4.	1 1 1 1 1 1		6

	PART C			
III(a)	AVR Status Register AVR Status Register is an 8 bit register. It is also referred to as flag register.  C – Carry flag S – Sign flag Z – Zero flag H – Half carry N – Negative flag T – Bit copy storage V – Overflow flag I – Global Interrupt Enable C, Carry flag This flag is set whenever there is a carry out from the D7th bit. This flag bit is affected after an 8 bit addition or subtraction. Z, The Zero flag The zero flag reflects the result of an arithmetic or logic operation. If the result is zero, then z=1.Therefore, Z=0 if the result is not zero. N, the negative flag Binary representation of signed number uses D7as the sign bit. If the D7 bit of the result is zero, then N=0 the result is positive. If the D7 bit is one, then N=1 and the result is negative. V, the overflow flag This flag is set whenever the result of a signed number operation is too large, causing the higher order bit to overflow into the sign bit. Overflow flag is used to detect errors in signed arithmetic operations. S, the sign flag This flag is the result of Exclusive – ORing of N and V flags. H, Half carry flag If there is a carry from D3 to D4during an ADD or SUB operation, this bit is set; otherwise, it is cleared. This flag bit is used by instructions that perform BCD arithmetic. T, Bit copy storage Transfer bit used by BLD (Bit load) and BST (Bit store) instructions I, Global interrupt Enable.	2 1 1 1 1 1 1 1 1		10

III(b)	Data Format representation
	There are four ways to represent a byte of data in AVR assembler. The number can be in hex, binary, decimal, or ASCII formats.
	Hex numbers
	There are two ways to show hex numbers:
	1. Put 0x(or 0X) in front of the number like LDI R16,ox99
	2. Put $ in front of the number like LDI R22,$99
	Binary numbers
	There are only one way to show binary numbers in an AVR assembler:
	LDI R23,0b00100101
	Decimal numbers
	To indicate decimal numbers in an AVR assembler use the decimal and nothing before or after it.
	LDI R17,12
		1
		
		1
		
		1
		
		1
				5

	ASCII characters To represent ASCII data in an AVR assembler use single quotes as follows: LDI R22,'2' ; R22=00110010 or 32 in hex	1		
IV(a)	The AVR Data memory  **GPRs(General Purpose Registers)** GPRs uses 32 bytes of data memory in the address location $00-$1F. The GPRs are designated as R0 to R31 **I/O Memory** The I/O memory is dedicated to specific functions such as status registers, timers, serial communications, I/O ports, ADC ad so on. The AVR I/O memory is made of 8 bit registers. **Internal SRAM** The internal data SRAM known as **scratch pad** is used for storing data and parameters by AVR programmers. Each location in the scratch pad is 8-bit wide and can be accessed directly by its address. The size of SRAM can vary from chip to chip.	5 1 1 1		8
IV(b)	AVR microcontrollers are broadly classified into four families • Classic AVR • Mega AVR • Tiny AVR • Special purpose AVR **Classic AVR (AT90Sxxxx)** Some members of classic family • AT90S2313 • AT90S2323 • AT90S4433	1 1.5		

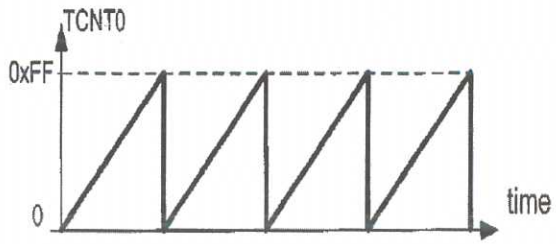
	Mega AVR (ATmegaxxxx) These are powerful microcontrollers with more than 120 instructions and lots of different peripheral capabilities. Some features are: program memory : 4K to 256K Package : 28 to 100 pins Extensive peripheral set Extended or rich instruction set • ATmega8 • ATmega16 • ATmega32 • ATmega64 • ATmega1280 **Tiny AVR (ATtinyxxxx)** The microcontrollers in this group have less instructions and smaller packages in comparison to mega family. Tiny AVRs can be used for the design of systems with low costs and power consumption. Important characteristics are: Program memory : 1K to 8K Package : 8 to 28 pins Limited peripheral cost Limited Instruction set • ATtiny13 • ATtiny25 • ATtiny44 • ATtiny84 **Special Purpose AVR** This family of microcontrollers are characterized with some special capabilities such as USB controller, CAN controller, LCD controller, Ethernet controller, FPGA and advanced PWM. They are used for designing specific application. • AT90CAN128 • AT90USB1287 • AT90PWM216	1.5 1.5 1.5		7

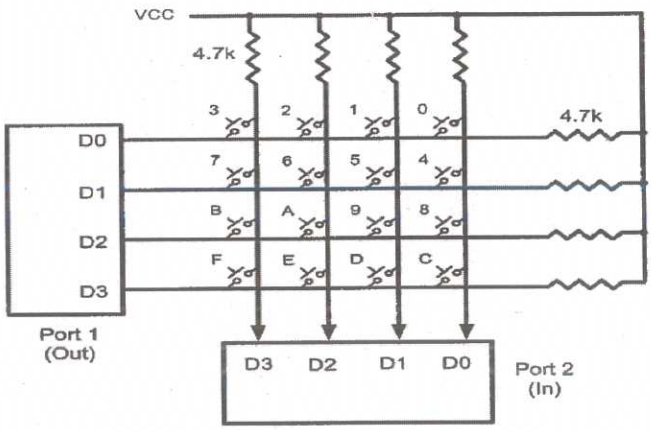
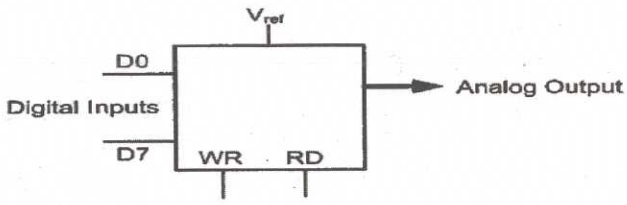
V(a)	#include<avr/io.h>
	int main(void)
	{
	DDRC=0;
	DDRB=0xFF;
	DDRD=0xFF;
	unsigned char temp;
	while(1)
	{
	temp=PINC;
	if(temp<100)
		1
		1
		1
		1
		1
		1
		1
		1
		1
				10

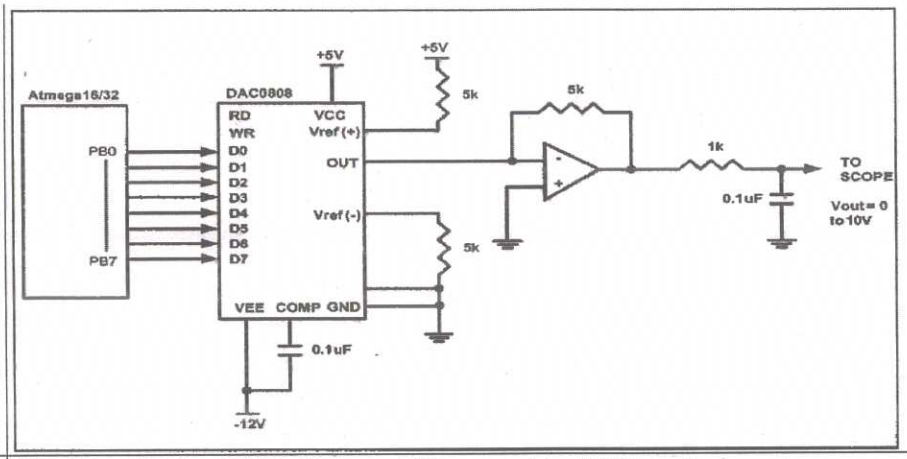
	PORTB=temp;
	else
	PORTD=temp;
	}
	return 0;
	}
		1
		
		1		

V(b)	The following are the major reasons for writing programs in C instead of Assembly 1. It is easier and less time consuming to write in C than in Assembly. 2. C is easier to modify and update. 3. You can use code available in function libraries. 4. C code is portable to other microcontrollers with little or no modification.	1 1 1 2		5
VI(a)	1. SBI 2. CBI 3. SBIC 4. SBIS SBI ioreg,bit-- Set bit in I/O register CBI ioreg,bit-- Clear bit in I/O register SBIC ioreg,bit--Skip if Bit in I/O register Cleared (skip next instruction if bit=0) SBIS ioreg,bit--Skip if Bit in I/O register Set(skip next instruction if bit=1) write example for each	1 1 1 1 1x4		9

VI(b)	#include<avr/io.h>
	int main(void)
	{
	unsigned char z;
	DDRB=0xFF;
	For(z=0;z<=255;z++)
	{
	PORTB=z;
	}
	return 0;
	}
		1
		1
		1
		1
		1
		1
				6

VII(a)	1. Bit D7(I) of the SREG register must be set to HIGH to allow the interrupts to happen. This is done with the SEI(Set Interrupt) instruction. 2. If I=1, each interrupt is enabled by setting t0 HIGH the Interrupt Enable(IE)flag bit for that interrupt. TIMSK register has interrupt enable bits. If I=0,no interrupt will be responded. LDI R20,(1<<TOIE0) OUT TIMSK,R20 SEI	3 2 2		7
VII(b)	The most widely used sources of interrupts in he AVR: 1. Atleast two interrupts are set aside for timer0,timer1,timer2, one for overflow and another for compare match. 2. Three interrupts are set aside for external hardware interrupts. Pins PD2(PORTD.2),PD3(PORTD.3), and PB2(PORTB.2) are for the external hardware interrupts INT0,INT1, and INT2 respectively. 3. Serial communication's USART has three interrupts, one for receive and two interrupts for transmit. 4. The SPI interrupts. 5. The ADC(Analog-to-digital coverter).	2 2 2 1 1		8
VIII(a)	TCCR1A Timer/Counter Control Registers TCCR1B TCNT1 Timer/Counter Count Register TOV1 Timer Overflow Flag OCR1A Output Compare Registers OCR1B OCF1A Output Compare Flag OCF1B	1 1 1 1 1		5
VIII(b)	Timer0 normal mode:-In this mode contents of the timer/counter increments with each clock. It counts up until it reaches its maximum of 0xFF. When it rolls over from 0xFF to 0x00, it sets high a flag bit called TOV0(Timer Overflow).  Steps to program Timer0 in Normal mode 1. Load the TCNT0 register with the initial count value. 2. Load the value into the TCCR0 register, indicating mode(8-bit or 16-bit) is to be used and the prescaler option. When select the clock source, the timer/ counter starts to count, and each tick causes the content of the timer/counter to increment by 1.	2 2		

	3. Keep monitoring the timer overflow flag(TOV0) to see if it is raised. Get out of the loop when TOV0 becomes high. 4. Stop the timer by disconnecting the clock source ,using the following instructions: LDI R20,0x00 OUT TCCR0,R20 ;timer stopped mode=normal 5. Clear the TOV0 flag for the next round 6. Go back to step1 to load TCNT0 again.	6		10
IX(a)	When a key is pressed, a row and a column make a contact; otherwise there is no connection between rows and columns.  Scanning and identifying the key The 4x4 matrix connected to two ports. The rows are connected to an output port and the columns are connected to an input port. If no key has been pressed, reading the input port will yield 1s for all columns since they are all connected to high(VCC). If all rows are grounded and a key is pressed, one of the column will have 0 since the key pressed provides path to ground.	1 3 3		7
IX(b)	DAC convert digital pulses to analog signals. The number of data bit inputs decides the resolution of the DAC because the number of analog output levels is equal to 2^n , where n is the number of data bit inputs.  An 8- input DAC0808 provides 256 discrete voltage levels of output. Digital inputs are converted to current (Iout) and by connecting a resister to the Iout pin convert the result to voltage. The total current provided by the Iout pin is a function of the binary numbers at the D0-D7 inputs Iout=Iref (D7/2 +D6/4 +D5/8 +D4/16+D3/32+D2/64+D1/128+D0/256)	3 2		



3

8

X(a) **Vcc,Vss,VEE**: While Vcc and Vss provide +5 V and ground ,respectively, VEE is used for controlling LCD contrast.

RS, register select: if RS=0, the instruction command code register is selected, allowing the user to send commands such as clear display, cursor at home ,and so on. If RS=1 the data register is selected, allowing the user to send data to be displayed on the LCD.

R/W, read/write: R/W input allows the user to write information to the LCD or read information from it. R?W=1 when reading; T/W=0 when writing.

E, enable: When data is supplied to data pins,a high to low pulse must be applied ti this pin in order for the LCD to latch in the data present at the data pins.

D0-D7; The 8-bit data pins, D0-D7 are used to send information to the LCD or read the contents of the LCD's internal registers.

Sending commands and data to the LCD

The procedure to send commands and data to the LCD is as follows:

1.Initialize the LCD

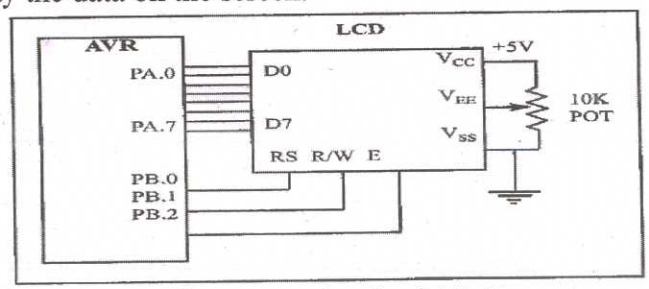
For 5x7 matrix and 8-bit operation, the following sequence of commands should be send to the LCD: 0x38, 0x0E and 0x01.

2. Send Any of the commands to the LCD.

To send any commands to the LCD, make pins RS and R/W=0 and put the command number on the data pins (D0-D7).Then send a high-to-low pulse to the E pin to enable the internal latch of the LCD. Wait for some time to let the LCD to execute the command.

3. Send the character to be shown on the LCD.

To send data to the LCD, make pins RS=1 R/W=0.Put the data on the data pins (D0-D7).Send a high-to-low pulse to the E pin to enable the internal latch of the LCD. Wait for some time to let the LCD module to display the data on the screen.



LCD connection for 8-bit data

2
1
1
1
1
2
2
2
3

15