PES COLLEGE OF ENGINEERG MANDYA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AVR MICROCONTROLLER

(INTEGRATED)

LAB MANUAL (P21CS405)

Preface

AVR microcontroller is an electronic chip manufactured by Atmel, which has several advantages over other types of microcontroller.

We can understand microcontroller by comparing it with Personal Computer (PC), which has a motherboard inside it. In that motherboard a microprocessor (AMD, Intel chips) is used that provides the intelligence, EEPROM and RAM memories for interfacing to the system like serial ports, display interfaces and disk drivers. A microcontroller has all or most of these features built into a single chip, therefore it doesnôt require a motherboard and any other components.

AVR microcontroller comes in different configuration, some designed using surface mounting and some designed using hole mounting. It is available with 8-pins to 100-pins, any microcontroller with 64-pin or over is surface mount only.

Some mostly used AVR microcontrollers are:-

- ATmega8 microcontroller
- ATmega16 microcontroller
- ATmega32 microcontroller
- ATmega328 microcontroller

ATmega32-8 Bit AVR Microcontroller

The AVR microcontroller is based on the advanced Reduced Instruction Set Computer (RISC) architecture. ATmega32 microcontroller is a low power CMOS technology based controller. Due to RISC architecture AVR microcontroller can execute 1 million of instructions per second if cycle frequency is 1 MHz provided by crystal oscillator.

Consider some general features of ATmega32 microcontroller is:-

- 2 Kilo bytes of internal Static RAM
- 32 general purpose registers
- 32 Kilo bytes of in system self programmable flash program memory.
- 1024 bytes EEPROM
- Programmable serial USART
- 8 Channel, 10 bit ADC
- One 16-bit timer/counter with separate prescaler, compare mode and capture mode.
- Available in 40 pin DIP, 44-pad QFN/MLF and 44-lead QTFP
- Two 8-bit timers/counters with separate prescalers and compare modes
- 32 programmable I/O lines
- In system programming by on-chip boot program
- Master/slave SPI serial interface
- 4 PWM channels
- Programmable watch dog timer with separate on-chip oscillator.

# of Contents

Program no.	Problem set
1	Write a program to find greatest of three numbers.
2	Write a program to div two numbers.
3	Write a program to whether the given number is power of 2 or not.
4	Write a program to find the factorial of given positive number.
5	Write a program to accept two 8 bit numbers from PORTA and POTRB. Multiply two numbers and send the result to PORTC (lower byte) and PORTD (higher byte).
6	Write a program to monitor the bit 1 of PORTC. If set send óYô to PORTA else send óNô to PORTB.
7	Write a program to convert packed BCD to ASCII.
8	Write a program to add 10 bytes of data stored starting from $300. Store the sum in R21 and carry in R22. (Use direct addressing mode).
9	Write a program to count number of odd and even numbers among n bytes of data stored starting from $600 (Use indirect addressing mode).
10	Write c program to blink LED connected to 1st pin of PORTC with appropriate delay.
11	A switch is connected to PORTA. If it is pressed the led should glow. Write C program to perform the above said operation.
12	Write a program to display the given message on LCD.

ts and Software Installation Procedures

Atmel Studio is the integrated development environment from Atmel®. It provides you a modern and powerful environment for doing AVR® and ARM development.

System requirements

Supported operating systems

- Windows XP (x86) with Service Pack 3 - all editions except Starter Edition
- Windows Vista (x86 & x64) with Service Pack 1 - all editions except Starter Edition
- Windows 7 (x86 and x64)
- Windows 8 / 8.1 (x86 and x64)
- Windows Server 2003 (x86 & x64) with Service Pack 2
- Windows Server 2003 R2 (x86 and x64)
- Windows Server 2008 (x86 and x64) with Service Pack 2
- Windows Server 2008 R2 (x64)

Supported architectures

- 32-Bit (x86)
- 64-Bit (x64)

Hardware requirements

- Computer that has a 1.6GHz or faster processor
- RAM
- 1 GB RAM for x86
- 2 GB RAM for x64
- An additional 512 MB RAM if running in a Virtual Machine
- 4 GB of available hard disk space
- A minimum display resolution of 1024 x 768 or higher is recommended

Downloading and installing

- Download the latest Atmel Studio installer.
- Atmel Studio 6.2 can be run side by side with Atmel Studio 6.1, Atmel Studio 6.0, AVR Studio® 5.0 and 5.1. Uninstallation of previous versions is not required.
- Verify the hardware and software requirements from "System Requirements" section.
- Make sure you are logged on with Administrative privileges.
- Please save all your work before starting, because the installation might prompt you for a restart if required.

rdware devices.

onnected to the internet during installation to allow for

ŏ Double click the installer executable file. Please note that this might take some time to extract depending on H/W configuration.

ŏ Atmel Studio Prerequisites installation will start.

NOTE : If you have all the prerequisites already installed then this dialog will not be shown.

ŏ If .NET Framework 4.0 is not already installed, the installer will start the .NET Framework setup, Note that this does not include SP1.

ŏ Accept the license agreement and proceed through the installation. If the installer prompts for restart please do so. After restart the installation will start automatically.

ŏ If Visual Studio Isolated shell 2010 is not installed, the installer will start the Microsoft Visual Studio Isolated Shell (2010) Setup.

ŏ Accept the license agreement and proceed through the installation.

ŏ The Atmel USB Driver will install/upgrade the existing Jungo USB driver, and will also keep the existing hardware to work.

ŏ Accept the license agreement and proceed through the installation.

● *Note: If you have previous versions of Jungo USB driver then the installer will update them. The Atmel USB driver (Jungo USB Driver + SAM USB Driver) is fully compatible with its previous versions. So AVR Studio 4, Studio 32 and AVR Studio 5.0, 5.1 should continue to work with the updated driver without any issues.*

ŏ After this, Atmel Studio installation should start.

ŏ Click ñNextò to continue.

ŏ Accept the license agreement and continue.

ŏ Choose the ñDestination Directory to Installò and click ñNextò.

ŏ Review the summary and click ñInstallò.

ŏ The installation will copy all files and prompt to click ñFinishò.

ŏ At the end, the installer will display options to associate files with AVR Studio. Please choose them if you prefer to open files with the mentioned file extensions in Atmel Studio.

ŏ The installer also displays an option to "Start Atmel Studio after completion". If you choose to open, then please note that Atmel Studio will launch with administrative privileges, since the installer was either launched as administrator or with elevated privileges (in Windows Vista or later).

Syllabus

Course Title : Object Oriented Programming with Java Laboratory			
Course Code : P18CSL48	Semester : 4	L :T:P:H : 0:0:3:3	Credits: 1.5
Contact Period: Laboratory : 3 Hrs/week, Exam: 3 Hr		Weightage: CIE:50%, SEE:50%	

Lab Set Programs

Unit	Program no.	Problem set
Unit-2	1	Write a program to find greatest of three numbers.
	2	Write a program to div two numbers.
	3	Write a program to whether the given number is power of 2 or not.
Unit-3	4	Write a program to find the factorial of given positive number.
	5	Write a program to accept two 8 bit numbers from PORTA and POTRB. Multiply two numbers and send the result to PORTC (lower byte) and PORTD (higher byte).
	6	Write a program to monitor the bit 1 of PORTC. If set send óYôto PORTA else send óNôto PORTB.
Unit-4	7	Write a program to convert packed BCD to ASCII.
	8	Write a program to add 10 bytes of data stored starting from $300. Store the sum in R21 and carry in R22. (Use direct addressing mode).
	9	Write a program to count number of odd and even numbers among n bytes of data stored starting from $600 (Use indirect addressing mode).
Unit-5	10	Write c program to blink LED connected to 1st pin of PORTC with appropriate delay.
	11	A switch is connected to PORTA. If it is pressed the led should glow. Write C program to perform the above said operation.
	12	Write a program to display the given message on LCD.

Course Outcomes The student is able to

CO1: Implement and **demonstrate** the concept identified in the co-course.

Course Articulation Matrix (CAM)														
Course Outcomes	Program Outcomes (PO's)												PSO's	
	1	2	3	4	5	6	7	8	9	10	11	12	1	2
CO 1	2	2	2		2				1			1	1	

of three numbers.

```
) (all numbers are decimal)
Output: 1B(Dec 27) at Memory Location 0x0060
*/
 .org 0
 .equ largest=0x0060
 ; load r16,r17 and r18 with three numbers
        ldi r16,-27
        ldi r17,27
        ldi r18,-10
 ;logic to find larsest number and r16 contains the largest number
        cp r16,r17
        brge l1
        mov r16,r17
  l1:   cp r16,r18
        brge l2
        mov r16,r18
  ; store the largest number in memory location
  l2:   sts largest,r16
here:   jmp here
```

ide two numbers.

Example- Input: 117=-10, 118=2 (all numbers are decimal)
Output: 5(result will be in Hexadecimal) at Memory Location 0x0300
*/

```
.org 0
.equ        result   =0X0300      ;address of result
.def        dividend =r17         ;dividend
.def        divisor =r18          ;divisor
.def        counter =r19          ;loop counter

ldi dividend,10
ldi divisor,2
ldi counter,0
clc
 l1:inc counter
 sub dividend,divisor
 brcc l1
 dec counter

; store the result in memory location

 sts result,counter
Here: rjmp Here
```

whether the given number is power of 2 or not.

Example- Input: R_{16}=128 (decimal)
Output: if the content of register R_{19} is 1, then the given number is power of 2 otherwise not.
*/

```
.org 0
 ldi r16,128
 ldi r19,0
  ldi r17,0x08        ;counter for loop
 clc
; right shift the content of register R16 8 times and count the number of 1ôs
 top: lsr r16
 brcc l1
 inc r19
 l1:  dec r17
       brne top
here:rjmp here
```

OR

/*
Example- Input: R_{16}=128 (decimal)
Output: Finally, if the content of register R_{17} is 0(zero) then the given number is power of 2, otherwise not.
*/

```
.org 0
 ldi r16,128         ;load the number into r16
mov r17,16           ;copy the content of r16 to r17
subi r17,1           ;find the previous number of the given number
andi r17,r16         ; perform logical AND on r17 and r16
here:rjmp here
```

f a number.

```
 Output:porta=0x78
*/
.org 0
;stack initialization
ldi r18,high(ramend)
out sph,r18
ldi r18,low(ramend)
out spl,r18
ldi r19,0x05
ldi r18,0x01
call fact          ;call subroutine
;make PORTA as output port to send result
ldi r16,0xff
out ddra,r16
out porta,r18
here: rjmp here

;subroutine
                    fact:   nop
                    loop:   mul r18,r19
                            mov r18,r0
                            dec r19
                            brne loop
                            ret     ;return to main code
```

accept two 8 bit numbers from PORTA and POTRB.
ult to PORTC (lower byte) and PORTD (higher byte).

```
Example Input: PORTA=0x45 and PORTB=0x20
Output: PORTD=0x08(higher byte) and PORTC=0xA0(lower byte)
*/

.org 0
 ;configure  PORTA,PORTB,PORTC and PORTD  as output port
 ldi r16,0xff
 out ddra,r16
 out ddrb,r16
out ddrc,r16
out ddrd,r16

 ;load the forst number to PORTA and second number to PORTB
ldi r16,0x45
 out porta,r16
ldi r17,0x20
 out portb,r17

 ;configure  PORTA and PORTB as input port
 ldi r20,0x00
 out ddra,r20
out ddrb,r20

 ;read the two numbers  from PINA and PINB
 in r24,pina
in r25,pinb

;multiply two numbers
mul r24,r25

;send result to PORT C and PORTD
out portc,r0
out portd,r1
here:rjmp here
```

itor the bit 1 of PORTC. If set send 'Y' to PORTA else

```
 Example 1- Input: PORTC=0x20(00100000)
 Output: PORTB=0x4E(ASCII for ôNõ)
Example 2- Input: PORTC=0x23(00100011)
 Output: PORTB=0x59(ASCII for ôYõ)

*/

.org 0

 ;configure  PORTA,PORTB  and  PORTC  as output port
 ldi r16,0XFF
 out ddra,r16
 out ddrb,r16
 out ddrc,r16

;load some value into PORTC
 ldi r17,0X20
 out portc,r17

;configure pin 1 of PORTC as Input pin
 cbi ddrc,1

;skip next instruction if PIN 1 of PORTC is set
sbis pinc,1
 rjmp over

;send ôYô to PORTA
 ldi r16,'Y'
 out porta,r16
 rjmp here

;send ôNô to PORTB
 over: ldi r16,'N'
 out portb,r16
here:  rjmp here
```

rt packed BCD to ASCII.

```
Example-Input:r20=0x48
   Output: memory location 0x301 = 0x34 and 0x302=0x38
*/
    .equ packedbcd=0x300
    .equ ascii1=0x301
    .equ ascii2=0x302

    ;initialize stack
    ldi r16,high(ramend)
    out sph,r16
    ldi r16,low(ramend)
    out spl,r16
    ;store packed BCD at location 0x300
    ldi r20,0x38
    sts packedbcd,r20
    call conversion    ;call subroutine
    here:jmp here

    ;subroutine
    Conversion:
          lds r20,packedbcd      ;load packedBCD into R20
            mov r21,r20
            ldi r22,0x30
            andi r20,0x0f        ;unpack BCD and obtain lowerbyte number
            add r20,r22          ;convert unpacked BCD into ASCII
            sts ascii1,r20       ;store first ASCII value into memory locatio 0x301
            swap r21
            andi r21,0x0f        ;unpack BCD and obtain higher byte number
            add r21,r22          ;convert unpacked BCD into ASCII
            sts ascii2,r21       ;store first ASCII value into memory locatio 0x302
            ret                  ;return to main program
```

l 10 bytes of data stored starting from $300. Store the addressing mode).

```
/*
Example: 0x300=10, 0x301=20, 0x302=30, 0x303=40, 0x304=50, 0x305=60, 0x306=70, 0x307=80,
0x308=90 and 0x309=100.(all numbers are in decimal)
Output: R21=0x26() and R22=1(carry value)
*/
    .include "m32def.inc"
     .org 0
    ; store 10 bytes of data starting from location 0x300 to 0x309
    Ldi r16,10
    Sts 0X0300,r16
    Ldi r16,20
    Sts 0X0301,r16
    Ldi r16,30
    Sts 0X0302,r16
    Ldi r16,40
    Sts 0X0303,r16
    Ldi r16,50
    Sts 0X0304,r16
    Ldi r16,60
    Sts 0X0305,r16
    Ldi r16,70
    Sts 0X0306,r16
    Ldi r16,80
    Sts 0X0307,r16
    Ldi r16,90
    Sts 0X0308,r16
    Ldi r16,100
    Sts 0X0309,r16

    Ldi r21,0
    Ldi r22,0

    ;read each byte from location 0x300 to 0x309 and add each value to R21
    Lds r20,0X0300
    Add r21,r20
    Lds r20,0x0301
    Add r21,r20
    Lds r20,0x0302
    Add r21,r20
    Lds r20,0x0303
    Add r21,r20
    Lds r20,0x304
    Add r21,r20
    Lds r20,0x0305
    Add r21,r20
    Lds r20,0x0306
```

```
Lds r20,0x0308
Add r21,r20
Lds r20,0x0309
Add r21,r20
;read the content of status register
in r19,sreg
lsr r19              ;obtain the content of carry bit
brcc Here
ldi r22,1            ;store the carry bit value into R22
Here:rjmp Here
```

t number of odd and even numbers among n bytes of ct addressing mode).

```
/
Example-  Input:if n=6, 0x600=2, 0x601=3, 0x602=4, 0x603=7, 0x604=10 and 0x605=12
Output: R20 = 2 (odd numbers) and R21 = 4 (even numbers)
Note : n value can be anything
*/
.org 0
ldi r16,10        ;counter(n)
ldi xl, 0X00      ;lower byte of address
ldi xh, 0X06      ;higher byte of address
ldi r20,0         ;odd number counter
ldi r21,0         ;even number counter

;store n=6 bytes of data from memory location 0x600 to 0x605
ldi r17,2
st x+,r17
ldi r17,3
st x+,r17
ldi r17,4
st x+,r17
ldi r17,7
st x+,r17
ldi r17,10
st x+,r17
ldi r17,12
st x+,r17

;initialize the register x with address 0x600
ldi xl, 0X00      ;lower byte of address
ldi xh, 0X06      ;higher byte of address


;obtain each byte from memory location 0x600 to 0x605
l1: ld r18,x+
   ror r18              ;obtain the last bit of each byte
brcs l2                 ;jump to l2 if the last bit is 1
inc r21                 ;increment odd counter
rjmp next
l2: inc r20             ;increment even counter
next: dec r16
brne l1
here: rjmp here
```

connected to 1st pin of PORTC with appropriate delay.

```
#define led_pin 8
void setup() {
pinMode(led_pin, OUTPUT);
}

void loop() {
  digitalWrite(led_pin, HIGH); //turn the led ON
delay(1000); // wait a second
digitalWrite(led_pin, LOW); //turn the led OFF
delay(1000); // wait a second
}
```

PORTA. If it is pressed the led should glow. Write C tion.

```
#define ledPin 13 // choose the pin for the LED
#define switchPin 7 // choose the input pin (for a pushbutton)

int val = 0; // variable for reading the pin status
void setup()
{
  pinMode(ledPin, OUTPUT); // declare LED as output
  pinMode(switchPin, INPUT); // declare pushbutton as input
}
void loop()
{
  val = digitalRead(switchPin); // read input value
  if (val == HIGH)
  { // check if the input is HIGH (button released)
    digitalWrite(ledPin, LOW); // turn LED OFF
  } else {
    digitalWrite(ledPin, HIGH); // turn LED ON } }
  }
}
```

he given message on LCD.

```
/*
  LiquidCrystal Library - Hello World

  Demonstrates the use a 16x2 LCD display.  The Liquid Crystal library works with all LCD
displays that are compatible with the  Hitachi HD44780 driver. There are many of them out there,
and you  can usually tell them by the 16-pin interface.

  This sketch prints "Hello World!" to the LCD  and shows the time.

   The circuit:
  * LCD RS pin to digital pin 12
  * LCD Enable pin to digital pin 11
  * LCD D4 pin to digital pin 5
  * LCD D5 pin to digital pin 4
  * LCD D6 pin to digital pin 3
  * LCD D7 pin to digital pin 2
  * LCD R/W pin to ground
  * LCD VSS pin to ground
  * LCD VCC pin to 5V
*/

  // include the library code:
  #include <LiquidCrystal.h>

  // initialize the library by associating any needed LCD interface pin
  // with the arduino pin number it is connected to
  const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
  LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

  void setup() {
    // set up the LCD's number of columns and rows:
    lcd.begin(16, 2);
    // Print a message to the LCD.
    lcd.print("hello, world!");
  }
```

```
    // (note: line 1 is the second row, since counting begins with 0):
    lcd.setCursor(0, 1);
    // print the number of seconds since reset:
    lcd.print(millis() / 1000);
}
```