ABSTRACT

The spark timing plays a significant role in the combustion process and in deciding the engine parameters of SI engine. This project aims at demonstrating the effect of advanced and retarded spark timing on the pressure variation as a function of crank angle with the help of ***Java*** programs. For this purpose, a basic finite heat release model is used for the combustion process in SI engines. This model can also be extended to evaluate effect of spark timing on engine work and thermal efficiency. In each section of the project, the codes used for analysis are provided for future research work

1. INTRODUCTION

Ignition timing is very important for improving performance of modern SI Engines. In an ideal four stroke SI engine, compression and expansion take place during 180° of crank rotation and combustion takes place instantaneously at TDC. During combustion the volume remains constant and there is a sudden pressure rise. However, in an actual spark ignition engine, combustion does not occur instantaneously. It is initiated by a spark produced before TDC at a definite time which affects the maximum pressure generated and the corresponding crank angle, indicated mean effective pressure, and consequentially the work done in cycle and so the thermal efficiency. Studies have been conducted to demonstrate some of these effects previously. This project essentially aims at generating ***Java*** programs which will take input of spark timing and will give output of the pressure developed inside the combustion chamber so that salient conclusions can be drawn. Computer models of engine processes are important tools for analysis of engine performance

2. MASTER PROGRAM

As mentioned earlier, in this project ***Java*** codes are provided for further research work, which can be applied to any SI engine. So a master program is first written which will be used to input the engine data for every individual code. This is the function where the spark timing will be entered along with other engine specific data.

a) code for **Velocity**(v) *v/s* **crank angle**(θ):

```
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.data.category.DefaultCategoryDataset;
import org.jfree.ui.ApplicationFrame;
import org.jfree.ui.RefineryUtilities;


class LineGraph_AWT extends ApplicationFrame{

   public LineGraph_AWT( String applicationTitle , String chartTitle )
   {
      super(applicationTitle);
      JFreeChart lineChart = ChartFactory.createLineChart(
            chartTitle,
            "Crank angle","Velocity",
            createDataset(0.1,22.0,.2),
            PlotOrientation.VERTICAL,
            true,true,false);

      ChartPanel chartPanel = new ChartPanel( lineChart );
      chartPanel.setPreferredSize( new java.awt.Dimension( 560 , 367 ) );
      setContentPane( chartPanel );
   }

   private DefaultCategoryDataset createDataset(double A, double w, double l){

      DefaultCategoryDataset dataset=new DefaultCategoryDataset();
      double[] v=new double[361];

      for(int j=0;j<361;j++){
         v[j]=A*w*(Math.sin(Math.PI*j/180)+(A/(2*l))*(Math.sin(2*Math.PI*j/180)));
         System.out.println(v[j]);
         dataset.addValue(v[j],"m/s",Integer.toString(j));
      }
      return dataset;
   }
```

```
  public static void main(String[] args){
    LineGraph_AWT chart = new LineGraph_AWT(
         "Velocity Graph" ,
         "Velocity vs Crank angle");

    chart.pack( );
    RefineryUtilities.centerFrameOnScreen( chart );
    chart.setVisible( true );
  }
```

b) code for **Acceleration**(*a*) v/s **crank angle**(ϴ):

```
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.data.category.DefaultCategoryDataset;
import org.jfree.ui.ApplicationFrame;
import org.jfree.ui.RefineryUtilities;


class LineGraph_AWT extends ApplicationFrame{

  public LineGraph_AWT( String applicationTitle , String chartTitle )
  {
    super(applicationTitle);
    JFreeChart lineChart = ChartFactory.createLineChart(
         chartTitle,
         "Crank angle","Acceleration",
         createDataset(0.1,22.0,.2),
         PlotOrientation.VERTICAL,
         true,true,false);

    ChartPanel chartPanel = new ChartPanel( lineChart );
    chartPanel.setPreferredSize( new java.awt.Dimension( 560 , 367 ) );
    setContentPane( chartPanel );
  }

  private DefaultCategoryDataset createDataset(double A, double w, double l){

    DefaultCategoryDataset dataset=new DefaultCategoryDataset();
    double[] a=new double[361];

    for(int j=0;j<361;j++){
      a[j]=A*w*w*(Math.cos(Math.PI*j/180)+(A/l)*(Math.cos(2*Math.PI*j/180)));
      System.out.println(a[j]);
      dataset.addValue(a[j],"m/s2",Integer.toString(j));
    }
    return dataset;
    }

    public static void main(String[] args){
        LineGraph_AWT chart = new LineGraph_AWT(
                "Acceleration Graph" ,
                "Acceleration vs Crank angle");

        chart.pack( );
        RefineryUtilities.centerFrameOnScreen( chart );
        chart.setVisible( true );
    }
}
```

c) code for **Pressure**(*p*) v/s **crank angle**(*θ*) :

```
import org.jfree.chart.ChartPanel;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.JFreeChart;
import org.jfree.ui.ApplicationFrame;
import org.jfree.ui.RefineryUtilities;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.data.category.DefaultCategoryDataset;

public class Main{

    double theta_s, theta_d, gamma, bore, stroke,con_rod, p0, Q,r,R;


    Main(double Theta_s,double Theta_d,double Gamma,double Bore,double Stroke,double
Con_rod,double P0,double Qin,double com_ratio){

        theta_s=Math.toRadians(Theta_s);
        theta_d=Math.toRadians(Theta_d);
        gamma=Gamma;
        bore=Bore;
        stroke=Stroke;
        con_rod=Con_rod;
        p0=P0;
        Q=Qin;
        r=com_ratio;

        R=2*con_rod/stroke;

    }

    Main(double Theta_s,double Theta_d){
        theta_s=Theta_s;
        theta_d=Theta_d;
    }
    public double getVd(){
        double Vd=(Math.PI)*bore*bore*stroke/4;
        //System.out.println("Vd="+Vd);
        return Vd;
    }

    public double getXb(double theta){
        double Xb=1-Math.exp(-5*Math.pow((theta-theta_s)/(theta_d),3));
        //System.out.println("Xb="+Xb);
        return Xb;
    }

    public double getV(double theta){
        double V
=(getVd()/(r-1))+(getVd()/2)*(R+1-Math.cos(theta)-Math.sqrt(R*R-Math.pow(Math.sin(thet
a),2)));
        //double V=(1+(r-1)/2*(1-Math.cos(Math.toRadians(theta))))/r;
        //System.out.println("V="+V);
        return V;
    }

    public double getdV(double theta){
        double dV
=(getVd()/2)*Math.sin(theta)*(1+Math.cos(theta)/Math.sqrt(R*R-Math.pow(Math.sin(theta),
2)));
        //double dV=((r-1)/2)*Math.sin(Math.toRadians(theta))/r;
        //System.out.println("dV="+dV);
        return dV;
    }

    public double getdQ(double theta){
        double dQ=15*(Q/theta_d)*(1-getXb(theta))*Math.pow((theta-theta_s)/(theta_d),2);
        //System.out.println("dQ="+dQ);
        return dQ;
    }

    public double f1(double theta,double p){
        double function=-1.4*p*getdV(theta)/getV(theta);
        return function;
    }

    public double f2(double theta,double p){
        double function=-1.4*p*getdV(theta)/getV(theta)+0.4*getdQ(theta)/getV(theta);
        return function;
    }

}
```

```
class LineChart_AWT extends ApplicationFrame
{
   public LineChart_AWT( String applicationTitle , String chartTitle )
   {
      super(applicationTitle);
      JFreeChart lineChart = ChartFactory.createLineChart(
            chartTitle,
            "Crank angle","Pressure",
            createDataset(),
            PlotOrientation.VERTICAL,
            true,true,false);

      ChartPanel chartPanel = new ChartPanel( lineChart );
      chartPanel.setPreferredSize( new java.awt.Dimension( 560 , 367 ) );
      setContentPane( chartPanel );
   }

   private DefaultCategoryDataset createDataset( ){
      Main test=new Main(-20,40,1.4,0.1,0.1,0.15,101325,1800,10);
      double p=101325;
      double k1,k2,k3,k4,p2;

      DefaultCategoryDataset dataset = new DefaultCategoryDataset( );
      dataset.addValue(p,"bar","-180");
      System.out.println(p);
      int i=-180;
      for(double
theta=-180*(Math.PI)/180;theta<=-20*(Math.PI)/180;theta=theta+2*(Math.PI)/180){

         k1=test.f1(theta,p);//function;
         //System.out.println("k1="+k1);
         k2=test.f1(theta+1*(Math.PI)/180,p+k1*(Math.PI)/180);
         //System.out.println("k2="+k2);
         k3=test.f1(theta+1*(Math.PI)/180,p+k2*(Math.PI)/180);
         //System.out.println("k3="+k3);
         k4=test.f1(theta+2*(Math.PI)/180,p+2*k3*(Math.PI)/180);
         //System.out.println("k4="+k4);

         p2=p+(k1+2*k2+2*k3+k4)*2*(Math.PI)/(180*6);

         p=p2;
         //System.out.println(theta*180/(Math.PI)+"="+p2);
         dataset.addValue(p2,"bar",Integer.toString(i+2));
         i=i+2;
         System.out.println(p2);
      }

      for(double
theta=-20*(Math.PI)/180;theta<=18*(Math.PI)/180;theta=theta+2*(Math.PI)/180){
         k1=test.f2(theta,p);//function;
```

```
            //System.out.println("k1="+k1);
            k2=test.f2(theta+1*(Math.PI)/180,p+k1*(Math.PI)/180);
            //System.out.println("k2="+k2);
            k3=test.f2(theta+1*(Math.PI)/180,p+k2*(Math.PI)/180);
            //System.out.println("k3="+k3);
            k4=test.f2(theta+2*(Math.PI)/180,p+2*k3*(Math.PI)/180);
            //System.out.println("k4="+k4);

            p2=p+(k1+2*k2+2*k3+k4)*2*(Math.PI)/(180*6);
            //p2=p+k1*;
            p=p2;

            dataset.addValue(p2,"bar",Integer.toString(i+2));
            //System.out.println(theta*180/(Math.PI)+"="+p2);

            i=i+2;
            System.out.println(p2);
         }
         for(double
   theta=20*(Math.PI)/180;theta<=180*(Math.PI)/180;theta=theta+2*(Math.PI)/180){

            k1=test.f1(theta,p);//function;
            //System.out.println("k1="+k1);
            k2=test.f1(theta+1*(Math.PI)/180,p+k1*(Math.PI)/180);
            //System.out.println("k2="+k2);
            k3=test.f1(theta+1*(Math.PI)/180,p+k2*(Math.PI)/180);
            //System.out.println("k3="+k3);
            k4=test.f1(theta+2*(Math.PI)/180,p+2*k3*(Math.PI)/180);
            //System.out.println("k4="+k4);

            p2=p+(k1+2*k2+2*k3+k4)*2*(Math.PI)/(180*6);

            p=p2;

            dataset.addValue(p2,"bar",Integer.toString(i+2));
            i=i+2;
            //System.out.println(theta*180/(Math.PI)+"="+p2);
            System.out.println(p2);
         }
         return  dataset;
      }
      public static void main( String[ ] args )
      {
         LineChart_AWT chart = new LineChart_AWT(
               "Pressure vs Crank angle" ,
               "Pressure vs Crank angle");

         chart.pack( );
         RefineryUtilities.centerFrameOnScreen( chart );
         chart.setVisible( true );
```

```
    }
}
```

3. FINITE HEAT RELEASE MODEL

Finite heat release model evaluates the effect of spark timing on heat release in combustion process and mathematically models, in differential form.
It is a differential form of a mathematical model of the power cycle in which the heat release is expressed as a function of the crank angle.

3.1 Burn Fraction Variation

A typical heat release curve consists of the initial ignition lag (flame development) followed by flame propagation and flame termination.
Weibe function or '*S*' curve describes the fraction of fuel that has been burned.
This function is represented mathematically

$$Xb(\Theta) = 1 - exp(- a\left((\Theta - \Theta s)/\Theta d\right)^{n}) \qquad \text{.... (1)}$$

Where,

$Xb(\Theta)$= Burn fraction as a function of crank angle
ϴ=crank angle
ϴs=spark timing
ϴd=Duration of heat release

n= Weibe form factor
a = Weibe efficiency factor

The boundary conditions for the model considered in code are that before ϴs the burn fraction will be zero and after (ϴs+ϴd) it will be one.

3.2 Heat Release Rate

In order to find the differential equation governing the heat release rate, eq.(1) is differentiated with respect to θ.

$$\frac{Xb(\Theta)}{d\Theta} =- exp(- a\left((\Theta - \Theta s)/\Theta d\right)^{n}\Big) \left(- an(\Theta - \Theta s)/\Theta d)^{n-1}\right). \frac{1}{\Theta d}$$

$$\frac{Xb(\Theta)}{d\Theta} = (1 - Xb). \frac{an}{\Theta d} . ((\Theta - \Theta s)/\Theta d)^{n-1}$$

The rate of heat release as a function of crank angle is

$$\frac{dQ}{d\Theta} = Qin . \frac{dXb}{d\Theta}$$

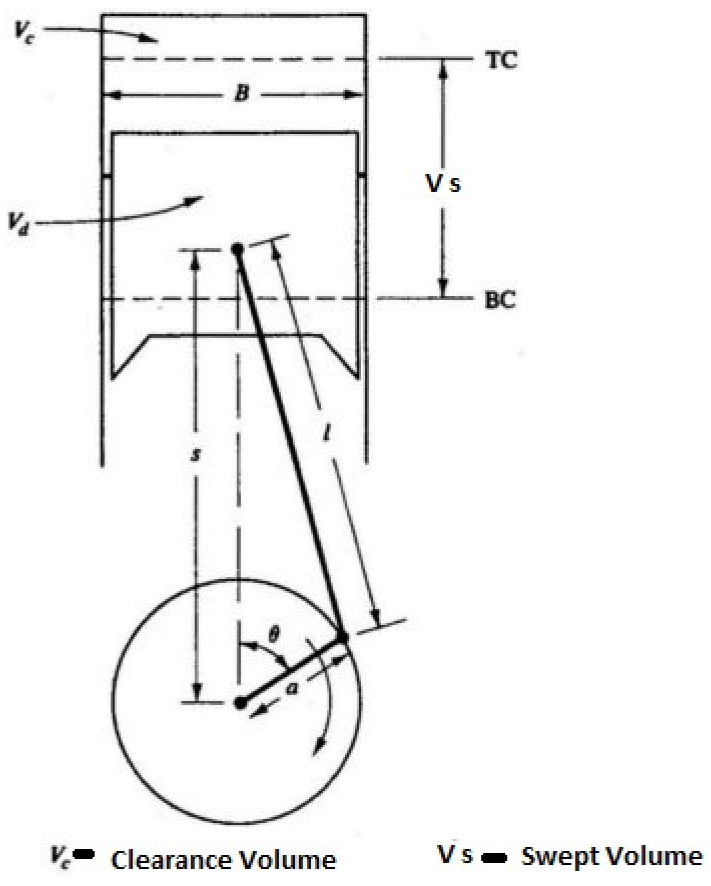
$$\frac{dQ}{d\Theta} = Qin . (1 - Xb). \frac{an}{\Theta d} . ((\Theta - \Theta s)/\Theta d)^{n-1} \qquad \text{... (2)}$$

Qin=heat supplied by the fuel.

When the combustion process is not happening this derivative is zero.

Thus, equation (2) gives the rate of heat release as a function of crank angle

3.3 Piston Volume Sweep



ϕ= angle which connecting rod makes with centreline

θ = crank angle from TDC position

a = crank radius

l= connecting rod length

from figure..

l sinϕ= a sinΘ

The distance s is given by

$$s = a\,cos\Theta + \sqrt{l^2 - a^2 sin^2\Theta}$$

Thus, the piston displacement from TDC is

$$x = l + a - s$$

$$x = l + a - a\,cos\Theta - \sqrt{l^2 - a^2 sin^2\Theta}$$

Instantaneous cylinder volume,

$$V = Vc + Vx$$

$$V = Vc + \frac{\pi}{4} d^2 x$$

$$V = \frac{Vs}{(r-1)} + \frac{Vs}{2}\left[R + 1 - cos\Theta - (R^2 - sin^2\Theta)^{\frac{1}{2}}\right] \quad ..(3)$$

Where,

r = compression ratio and

R = l/a

Vs = displacement Volume

Vc =clearance Volume

Now, differentiating above equation with respect to Θ ;

$$\frac{dV}{d\Theta} = \frac{Vs}{2}\sin sin\,\Theta\left[1 + cos\Theta\,(R^2 - sin^2\Theta)^{\frac{-1}{2}}\right] \quad ..(4)$$

3.4 Pressure - crank angle relation

From the first law of thermodynamics applied for a closed system ;

$$dQ = dU + dW$$

But , for reversible process;

$$dQ = pdV + mCv\,dT$$

From equation of state,

$$pV = mRT$$

$$pdV + vdP = mRdT \qquad ..(5)$$

$$mdT = \frac{pdV+vdP}{R} \qquad ..(6)$$

From equation (4) and (5);

$$dQ = pdV + \frac{R}{(\gamma-1)}\left(\frac{pdV+vdP}{R}\right)$$

$$dQ = pdV + \left(\frac{pdV+vdP}{(\gamma-1)}\right)$$

Differentiating with respect to Θ;

$$\frac{dQ}{d\Theta} = \frac{\gamma}{(\gamma-1)} \cdot \frac{pdV}{d\Theta} + \frac{1}{(\gamma-1)} \cdot \frac{Vdp}{d\Theta}$$

$$\frac{dp}{d\Theta} = \frac{(\gamma-1)}{V} \cdot \frac{dQ}{d\Theta} - \frac{p}{V} \cdot \frac{\gamma dV}{d\Theta} \qquad ..(7)$$

Equation (2) for $\frac{dQ}{d\Theta}$ and equation (4) for $\frac{dV}{d\Theta}$ and equation (3) for V are substituted in equation (7) ; the term is obtained as a function of crank angle Θ.

This term is integrated numerically using fourth order Runge-Kutta method for pressure at different crank angles using ***Java*** programming language.

4 RESULTS FROM FINITE HEAT RELEASE MODEL

Depending upon the spark timing the maximum pressure generated in the combustion process and the corresponding crank angle for maximum pressure varies.

Using the ***Java*** program as given in section 2 peak pressure has been noted with corresponding value of 'thetas' and engine parameters

Engine Parameters are:

Peak pressure is :

5. CONCLUSION AND FURTHER WORK

If the spark timing is over advanced, the combustion process starts while the piston is moving towards TDC, so the compression work (negative work) increases. On the other hand, if the spark timing is too much retarded, the combustion process is delayed, the peak pressure occurs much later in the expansion stroke.

The integration of equation (6) proceeds degree by degree to top dead center and back to bottom dead center. Thus, further work can be continued from this point to find out the net work done in the cycle from $dW = pdV$ and

and temperature from ideal gas law, $T = \frac{pV}{mR.}$

A computer code can be written to find out the effect of spark timing on net work done. Thermal efficiency and indicated mean effective pressure for each cycle can be determined. And thus, optimum spark timing will be the one at which maximum thermal efficiency will be obtained.

6. REFERENCES

[1] Fundamentals of Internal Combustion Engine - H.N Gupta , Second Edition PHI Learning

[2] Internal Combustion Engines by Ferguson and Kirkpatrick, Wiley 2001.