```
   Help
#include "cirpp1d_stdi.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2007+2) //The "#else" part of the code will be freely av
    ailable after the (year of creation of this file + 2)
int CALC(CF_ZCPutBondEuro)(void *Opt,void *Mod,Pricing
    Method *Met)
{
return AVAILABLE_IN_FULL_PREMIA;
}
static int CHK_OPT(CF_ZCPutBondEuro)(void *Opt, void *Mod)
{
  return NONACTIVE;
}
#else

/*Shift function of the CIR++ model*/
static double shift(double a,double b,double sigma,double
    f0_s,double s)
{
  /* the shift rate of the cir++ model for x(0)=0 */
  double  c;

  c=sqrt(a*a+2*sigma*sigma);

  return (f0_s - 2*a*b*(exp(s*c)-1)/(2*c+(a+c)*(exp(s*c)-1)
    ));
}

static double A(double time,double a,double b,double sigma)
{
  double h=sqrt(SQR(a)+2.*SQR(sigma));
  return pow(h*exp(0.5*(a+h)*(time))/(h+0.5*(a+h)*(exp(h*(
    time))-1.)),2.*a*b/SQR(sigma));
}

static double B(double time,double a,double b,double sigma)
{
  double h=sqrt(SQR(a)+2.*SQR(sigma));
  return (exp(h*(time))-1.)/(h+0.5*(a+h)*(exp(h*(time))-1.)
    );
}

/*Zero Coupon Bond*/
static double zcb_cirpp1d(double rcc,int flat_flag,double
    a,double b,double sigma,double t,double T, ZCMarketData*
    ZCMarket)
{
    double h, A, B, At, AT, shift, c;
    double f0_t,P0_t,P0_T;

    P0_t=BondPrice(t, ZCMarket);
    P0_T=BondPrice(T, ZCMarket);

    /*Computation of Forward rate*/
    if(t-0.5*INC>0)
    {
        f0_t = (log( BondPrice(t-0.5*INC, ZCMarket))-log(
    BondPrice(t+0.5*INC, ZCMarket)))/INC;
    }
    else
    {
        f0_t = -log( BondPrice(INC, ZCMarket))/INC;
    }

    /*A,B coefficient*/
    h=sqrt(SQR(a)+2.*SQR(sigma));
    B=2.*(exp(h*(T-t))-1.)/(2.*h+(a+h)*(exp(h*(T-t))-1.));
    A=pow(h*exp(0.5*(a+h)*(T-t))/(h+0.5*(a+h)*(exp(h*(T-t))
    -1.)), 2.*a*b/SQR(sigma));
    At=pow(h*exp(0.5*(a+h)*(t))/(h+0.5*(a+h)*(exp(h*(t))-1.
    )), 2.*a*b/SQR(sigma));
    AT=pow(h*exp(0.5*(a+h)*(T))/(h+0.5*(a+h)*(exp(h*(T))-1.
    )), 2.*a*b/SQR(sigma));

    c=sqrt(a*a+2*sigma*sigma);

    shift = (f0_t - 2*a*b*(exp(t*c)-1)/(2*c+(a+c)*(exp(t*c)
    -1)));

    A=A*(P0_T*At)/(AT*P0_t)*exp(B*shift);
    /*Price*/
    return A*exp(-B*rcc);

}

/*Call Option*/
static int zbp_cirpp1d(int flat_flag, double a, double b,
    double t,double sigma, double rcc, double S, double T,NumFunc_1
    *p,double *price,double *delta)
{
    double K;
    double PtS,PtT,ATS,BTS;
    double f0_t;
    double p1,p2,p3,k1,k2,k3,psi,phi,rb;
    double h;

    ZCMarketData ZCMarket;

    /* Flag to decide to read or not ZC bond datas in "ini
    tialyields.dat" */
    /* If P(0,T) not read then P(0,T)=exp(-r0*T) */
    if(flat_flag==0)
    {
        ZCMarket.FlatOrMarket = 0;
        ZCMarket.Rate = rcc;
    }

    else
    {
        ZCMarket.FlatOrMarket = 1;
        ReadMarketData(&ZCMarket);

        if(T > GET(ZCMarket.tm,ZCMarket.Nvalue-1))
        {
            printf("{nError : time bigger than the last
    time value entered in initialyield.dat{n");
            exit(EXIT_FAILURE);
        }
    }
    /*Computation of Forward rate*/
    h=sqrt(SQR(a)+2.*SQR(sigma));

    if(t-0.5*INC>0){f0_t = (log( BondPrice(t-0.5*INC, &ZCM
    arket))-log( BondPrice(t+0.5*INC, &ZCMarket)))/INC;}
    else {f0_t = -log( BondPrice(INC, &ZCMarket))/INC; }
    K=p->Par[0].Val.V_DOUBLE;
    PtT=zcb_cirpp1d(rcc,flat_flag,a,b,sigma,t,T, &ZCMarket)
    ;
    PtS=zcb_cirpp1d(rcc,flat_flag,a,b,sigma,t,S, &ZCMarket)
    ;

    BTS=B(S-T,a,b,sigma);
    ATS=A(S-T,a,b,sigma);

    /*X^2 parameters*/
    rb=(log(ATS/K)+log(A(T,a,b,sigma)*BondPrice(S, &ZCMarke
    t))-log(A(S,a,b,sigma)*BondPrice(T, &ZCMarket)))/BTS;
    if(rb<0){rb=0;}
    phi=2.*h/(SQR(sigma)*(exp(h*(T-t))-1.));
    psi=(a+h)/SQR(sigma);

    p1=2.*rb*(phi+psi+BTS);
    p2=4.*a*b/SQR(sigma);
    p3=2.*SQR(phi)*( rcc - shift(a,b,sigma,f0_t,t) )*exp(h*
    (T-t))/(phi+psi+BTS);

    k1=2.*rb*(phi+psi);
    k2=p2;
    k3=2.*SQR(phi)*( rcc - shift(a,b,sigma,f0_t,t) )*exp(h*
    (T-t))/(phi+psi);

    /*Price of Put by Parity*/
    K=p->Par[0].Val.V_DOUBLE;
    *price=PtS*pnl_cdfchi2n(p1,p2,p3)-K*PtT*pnl_cdfchi2n(k1
    ,k2,k3) - PtS+K*PtT;

    *delta=pnl_cdfchi2n(p1,p2,p3) - 1;

  return OK;
}
```

```
int CALC(CF_ZCPutBondEuro)(void *Opt,void *Mod,Pricing
    Method *Met)
{
  TYPEOPT* ptOpt=(TYPEOPT*)Opt;
  TYPEMOD* ptMod=(TYPEMOD*)Mod;

  return zbp_cirpp1d(ptMod->flat_flag.Val.V_INT,ptMod->a.
    Val.V_DOUBLE,ptMod->b.Val.V_DOUBLE,ptMod->T.Val.V_DATE,
                     ptMod->Sigma.Val.V_PDOUBLE,MOD(GetYi
    eld)(ptMod),ptOpt->BMaturity.Val.V_DATE,
                     ptOpt->OMaturity.Val.V_DATE,ptOpt->
    PayOff.Val.V_NUMFUNC_1,&(Met->Res[0].Val.V_DOUBLE),
                     &(Met->Res[1].Val.V_DOUBLE));
}

static int CHK_OPT(CF_ZCPutBondEuro)(void *Opt, void *Mod)
{
  return strcmp( ((Option*)Opt)->Name,"ZeroCouponPutBondEu
    ro");
}

#endif //PremiaCurrentVersion
static int MET(Init)(PricingMethod *Met,Option *Opt)
{
  if ( Met->init == 0)
    {
      Met->init=1;
    }

  return OK;
}

PricingMethod MET(CF_ZCPutBondEuro)=
{
  "CF_Cirpp1d_ZBPutEuro",
  {{" ",PREMIA_NULLTYPE,{0},FORBID}},
  CALC(CF_ZCPutBondEuro),
  {{"Price",DOUBLE,{100},FORBID},{"Delta",DOUBLE,{100},FORB
```

```
    ID} ,{" ",PREMIA_NULLTYPE,{0},FORBID}},
  CHK_OPT(CF_ZCPutBondEuro),
  CHK_ok,
  MET(Init)
} ;
```

References