

```
#include "sierrachart.h"
```

```
SCSFExport scsf_StochasticCrossover(SCStudyInterfaceRef sc)
{
    SCSubgraphRef Subgraph_Buy = sc.Subgraph[0];
    SCSubgraphRef Subgraph_Sell = sc.Subgraph[1];
    SCSubgraphRef Subgraph_FastD = sc.Subgraph[2];
    SCFloatArrayRef Subgraph_SlowK = Subgraph_FastD.Arrays[0];
    SCFloatArrayRef Subgraph_SlowD = Subgraph_FastD.Arrays[1];

    SCInputRef Input_FastK = sc.Input[0];
    SCInputRef Input_FastD = sc.Input[1];
    SCInputRef Input_SlowD = sc.Input[2];
    SCInputRef Input_Line1 = sc.Input[3];
    SCInputRef Input_Line2 = sc.Input[4];
    SCInputRef Input_UseBuySell = sc.Input[5];
    SCInputRef Input_ArrowOffsetPercentage = sc.Input[6];

    if (sc.SetDefaults)
    {
        // Set the configuration and defaults

        sc.GraphName = "Stochastic Crossover System";
        sc.StudyDescription = "This is an example of a Stochastic Crossover Study System.";

        Subgraph_Buy.Name = "Buy";
        Subgraph_Buy.PrimaryColor = RGB_COLOR(0, 255, 0); // green
        Subgraph_Buy.DrawStyle = DRAWSTYLE_ARROW_UP;
        Subgraph_Buy.LineWidth = 2; //Width of arrow

        Subgraph_Sell.Name = "Sell";
        Subgraph_Sell.DrawStyle = DRAWSTYLE_ARROW_DOWN;
        Subgraph_Sell.PrimaryColor = RGB_COLOR(255, 0, 0); // red
        Subgraph_Sell.LineWidth = 2; //Width of arrow

        Input_FastK.Name = "Fast %K Length";
        Input_FastK.SetInt(10);

        Input_FastD.Name = "Fast %D Length";
        Input_FastD.SetInt(3);

        Input_SlowD.Name = "Slow %D Length";
        Input_SlowD.SetInt(3);

        Input_Line1.Name = "Line 1 Value";
        Input_Line1.SetFloat(70);

        Input_Line2.Name = "Line 2 Value";
        Input_Line2.SetFloat(30);

        Input_UseBuySell.Name = "Use Buy/Sell Lines";
        Input_UseBuySell.SetYesNo(true);

        Input_ArrowOffsetPercentage.Name = "Arrow Offset Percentage";
        Input_ArrowOffsetPercentage.SetInt(3);

        sc.GraphRegion = 0; //Main chart region
        sc.DrawZeros= 0;

        sc.AutoLoop = 1;

        return;
    }
}
```

```

// Do data processing

// The following line calculates the stochastic.
// The 3 stochastic lines are placed in sc.Subgraph[2].Data, sc.Subgraph[2].Arrays[0] and sc.Subgraph[2].Arrays[1].
sc.Stochastic(sc.BaseDataIn, Subgraph_FastD, Input_FastK.GetInt(), Input_FastD.GetInt(), Input_SlowD.GetInt(),
MOVAVGTYPE_SIMPLE);

int Index=sc.Index; //Sets the variable Index to the current Bar index.

/* This is used further down to position a buy or sell arrow below or above a price bar so it does not touch exactly the
low or high. This is not necessary. It is only for appearance purposes.*/
float Offset=(sc.High[Index]-sc.Low[Index])*(Input_ArrowOffsetPercentage.GetInt()* 0.01f);

/*This code is not currently used:

//Calculate crossing point
float CrossY = 0.0f;
float p1x, p2x, p3x, p4x;
float p1y, p2y, p3y, p4y;

p1x = p3x = (float)(Index - 1);
p2x = p4x = (float)(Index);

if(SlowK[Index] > SlowD[Index])
{
    p1y = SlowD[Index - 1];
    p2y = SlowD[Index];
    p3y = SlowK[Index - 1];
    p4y = SlowK[Index];
}
else
{
    p1y = SlowK[Index - 1];
    p2y = SlowK[Index];
    p3y = SlowD[Index - 1];
    p4y = SlowD[Index];
}

float t
    = ((p4x - p3x) * (p1y - p3y) - (p4y - p3y) * (p1x - p3x)) /
    (float)((p4y - p3y) * (p2x - p1x) - (p4x - p3x) * (p2y - p1y));

CrossY = p1y + (t * (p2y - p1y));

*/

/*If Slow %k crosses Slow %d from the bottom AND %k is below 30*/
if (sc.CrossOver( Subgraph_SlowK, Subgraph_SlowD, Index) == CROSS_FROM_BOTTOM &&
    (!Input_UseBuySell.GetYesNo() || Subgraph_SlowK[Index] < Input_Line2.GetFloat()))
{

    //Place an Up arrow below the low of the current bar.
    Subgraph_Buy[Index] = sc.Low[Index] - Offset;

    /*Make sure we have no down arrow from a prior calculation. It is important to set this to zero since we perform
calculations on the same bar many times while the bar is being updated and we may have had a prior sell signal when we
now have a buy signal or no signal.*/
    Subgraph_Sell[Index] = 0;
}

/*If Slow %k crosses Slow %d from the top AND %k is over 70*/
else if (sc.CrossOver(Subgraph_SlowK, Subgraph_SlowD, Index) == CROSS_FROM_TOP &&
    (!Input_UseBuySell.GetYesNo() || Subgraph_SlowK[Index] > Input_Line1.GetFloat()))

```

```

{
    Subgraph_Sell[Index] = sc.High[Index] + Offset;
    Subgraph_Buy[Index] = 0;
}
else
{
    Subgraph_Buy[Index] = 0;
    Subgraph_Sell[Index] = 0;
}
}

/*****
SCSFExport scsf_MACDCrossoverSystem(SCStudyInterfaceRef sc)
{
    SCSubgraphRef Subgraph_Buy = sc.Subgraph[0];
    SCSubgraphRef Subgraph_Sell = sc.Subgraph[1];
    SCSubgraphRef Subgraph_MACDData = sc.Subgraph[2];

    SCInputRef Input_InputData = sc.Input[0];
    SCInputRef Input_FastLength = sc.Input[1];
    SCInputRef Input_SlowLength = sc.Input[2];
    SCInputRef Input_MACDLength = sc.Input[3];
    SCInputRef Input_MAType = sc.Input[4];
    SCInputRef Input_OffsetPercentInput = sc.Input[5];

    if (sc.SetDefaults)
    {
        // Set the configuration and defaults

        sc.GraphName = "MACD Crossover System";

        sc.GraphRegion = 0; //Main chart region

        sc.AutoLoop = 1;

        Subgraph_Buy.Name = "Buy";
        Subgraph_Buy.PrimaryColor = RGB_COLOR(0, 255, 0); // green
        Subgraph_Buy.DrawStyle = DRAWSTYLE_ARROW_UP;
        Subgraph_Buy.LineWidth = 2; //Width of arrow
        Subgraph_Buy.DrawZeros = 0;

        Subgraph_Sell.Name = "Sell";
        Subgraph_Sell.DrawStyle = DRAWSTYLE_ARROW_DOWN;
        Subgraph_Sell.PrimaryColor = RGB_COLOR(255, 0, 0); // red
        Subgraph_Sell.LineWidth = 2; //Width of arrow
        Subgraph_Sell.DrawZeros = 0;

        Input_InputData.Name = "Input Data";
        Input_InputData.SetInputDataIndex(SC_LAST);

        Input_FastLength.Name = "Fast Moving Average Length";
        Input_FastLength.SetInt(12);
        Input_FastLength.SetIntLimits(1, MAX_STUDY_LENGTH);

        Input_SlowLength.Name = "Slow Moving Average Length";
        Input_SlowLength.SetInt(26);
        Input_SlowLength.SetIntLimits(1, MAX_STUDY_LENGTH);

        Input_MACDLength.Name = "MACD Moving Average Length";
        Input_MACDLength.SetInt(9);

```

```

Input_MACDLength.SetIntLimits(1,MAX_STUDY_LENGTH);

Input_MAType.Name = "Moving Average Type";
Input_MAType.SetMovAvgType(MOAVGTYPE_EXPONENTIAL);

Input_OffsetPercentInput.Name = "Arrow Offset Percentage";
Input_OffsetPercentInput.SetFloat(8);

return;
}

// Do data processing
sc.DataStartIndex = max(Input_FastLength.GetInt(), Input_SlowLength.GetInt()) + Input_MACDLength.GetInt() - 1;

sc.MACD(sc.BaseDataIn[Input_InputData.GetInputDataIndex()], Subgraph_MACDData, sc.Index,
Input_FastLength.GetInt(), Input_SlowLength.GetInt(), Input_MACDLength.GetInt(), Input_MAType.GetInt());

if (sc.Index < sc.DataStartIndex)
return;

float Range = (sc.High[sc.Index] - sc.Low[sc.Index]);

float OffsetPercent = Input_OffsetPercentInput.GetFloat() * 0.01f;

SCFloatArrayRef MACDLine = Subgraph_MACDData.Data;
SCFloatArrayRef MACDMovingAverageLine = Subgraph_MACDData.Arrays[2];

if (sc.CrossOver( MACDLine, MACDMovingAverageLine) == CROSS_FROM_BOTTOM)
{
    Subgraph_Buy[sc.Index] = sc.Low[sc.Index] - OffsetPercent * Range;

    Subgraph_Sell[sc.Index] = 0;
}
else if (sc.CrossOver(Subgraph_MACDData, Subgraph_MACDData.Arrays[2]) == CROSS_FROM_TOP)
{
    Subgraph_Sell[sc.Index] = sc.High[sc.Index] + OffsetPercent * Range;
    Subgraph_Buy[sc.Index] = 0;
}
else
{
    Subgraph_Buy[sc.Index] = 0;
    Subgraph_Sell[sc.Index] = 0;
}
}

/*****

SCSFExport scsf_MACDZeroCrossOverSystem(SCStudyInterfaceRef sc)
{
    SCSubgraphRef Subgraph_Buy = sc.Subgraph[0];
    SCSubgraphRef Subgraph_Sell = sc.Subgraph[1];
    SCSubgraphRef Subgraph_MACDData = sc.Subgraph[2];

    SCInputRef Input_InputData = sc.Input[0];
    SCInputRef Input_FastLen = sc.Input[1];
    SCInputRef Input_SlowLen = sc.Input[2];
    SCInputRef Input_MACDLen = sc.Input[3];
    SCInputRef Input_MAType = sc.Input[4];
    SCInputRef Input_OffsetPercentInput = sc.Input[5];

```

```

if (sc.SetDefaults)
{
    // Set the configuration and defaults

    sc.GraphName = "MACD Zero Cross Over System";

    sc.GraphRegion = 0; //Main chart region

    sc.AutoLoop = 1;

    Subgraph_Buy.Name = "Buy";
    Subgraph_Buy.PrimaryColor = RGB_COLOR(0, 255, 0); // green
    Subgraph_Buy.DrawStyle = DRAWSTYLE_ARROW_UP;
    Subgraph_Buy.LineWidth = 2; //Width of arrow
    Subgraph_Buy.DrawZeros = 0;

    Subgraph_Sell.Name = "Sell";
    Subgraph_Sell.DrawStyle = DRAWSTYLE_ARROW_DOWN;
    Subgraph_Sell.PrimaryColor = RGB_COLOR(255, 0, 0); // red
    Subgraph_Sell.LineWidth = 2; //Width of arrow
    Subgraph_Sell.DrawZeros = 0;

    Input_InputData.Name = "Input Data";
    Input_InputData.SetInputDataIndex(SC_LAST);

    Input_FastLen.Name = "Fast Moving Average Length";
    Input_FastLen.SetInt(12);
    Input_FastLen.SetIntLimits(1, MAX_STUDY_LENGTH);

    Input_SlowLen.Name = "Slow Moving Average Length";
    Input_SlowLen.SetInt(26);
    Input_SlowLen.SetIntLimits(1, MAX_STUDY_LENGTH);

    Input_MACDLen.Name = "MACD Moving Average Length";
    Input_MACDLen.SetInt(9);
    Input_MACDLen.SetIntLimits(1, MAX_STUDY_LENGTH);

    Input_MAType.Name = "Moving Average Type";
    Input_MAType.SetMovAvgType(MOAVGTYPE_EXPONENTIAL);

    Input_OffsetPercentInput.Name = "Arrow Offset Percentage";
    Input_OffsetPercentInput.SetFloat(8);

    return;
}

// Do data processing
sc.MACD(sc.BaseDataIn[Input_InputData.GetInputDataIndex()], Subgraph_MACDData, sc.Index,
Input_FastLen.GetInt(), Input_SlowLen.GetInt(), Input_MACDLen.GetInt(), Input_MAType.GetInt());

float Range = (sc.High[sc.Index] - sc.Low[sc.Index]);

float OffsetPercent = Input_OffsetPercentInput.GetFloat() * 0.01f;

if (sc.CrossOver( Subgraph_MACDData.Arrays[8], Subgraph_MACDData.Arrays[2]) == CROSS_FROM_TOP)
{
    Subgraph_Buy[sc.Index] = sc.Low[sc.Index] - OffsetPercent * Range;

    Subgraph_Sell[sc.Index] = 0;
}
else if (sc.CrossOver( Subgraph_MACDData.Arrays[8], Subgraph_MACDData.Arrays[2]) ==

```

CROSS_FROM_BOTTOM)

```
{
    Subgraph_Sell[sc.Index] = sc.High[sc.Index] + OffsetPercent * Range;
    Subgraph_Buy[sc.Index] = 0;
}
else
{
    Subgraph_Buy[sc.Index] = 0;
    Subgraph_Sell[sc.Index] = 0;
}
}
```

/*****/