

// This Pine Script® code is subject to the terms of the Mozilla Public License 2.0 at <https://mozilla.org/MPL/2.0/> MPL-2.0

//@version=6

indicator('MFB - Trade Entry Quality', overlay = true, max_labels_count = 500)

// --- Constants ---

string NY_TIMEZONE = 'America/New_York'

string DESIGNED_TIMEFRAMES = 'Designed for 1m, 3m, 5m, 15m, 30m, and 1h only.'

// --- Inputs: Logic ---

primingLookbackInput = input.int(30, 'Level Touch Lookback', minval = 1, group = 'Logic Settings', tooltip = 'Number of bars a key-level touch remains valid for a regular CISD signal.')

atrLengthInput = input.int(14, 'ATR Length', minval = 1, group = 'Displacement Settings', tooltip = 'ATR lookback used to normalize each approach candle.')

approachLookbackInput = input.int(3, 'Approach Candle Count', minval = 1, maxval = 10, group = 'Displacement Settings', tooltip = 'Number of completed candles evaluated through the key-level touch candle.')

minQualifyingCandlesInput = input.int(2, 'Minimum Qualifying Candles', minval = 1, maxval = 10, group = 'Displacement Settings', tooltip = 'Minimum number of approach candles that must exceed the displacement and closure thresholds.')

displacementThresholdInput = input.float(1.2, 'Displacement Threshold (ATR)', minval = 0.1, step = 0.1, group = 'Displacement Settings', tooltip = 'A candle qualifies when its true range is at least this multiple of ATR.')

```
minClosureEfficiencyInput = input.float(0.65, 'Minimum Closure Efficiency',  
minval = 0.0, maxval = 1.0, step = 0.05, group = 'Displacement Settings',  
tooltip = 'Minimum directional close-location value. Higher values require  
closes nearer the approach direction.')
```

```
// --- Inputs: Display ---
```

```
showLevelsInput = input.bool(false, 'Show Key Levels', group = 'Display  
Settings', tooltip = 'Displays the PDH, PDL, PWH, and intraday session levels  
used by the MFB CISC rules.')
```

```
showEveningRangeInput = input.bool(true, 'Show 6pm-8pm Levels', group =  
'Display Settings', tooltip = 'Displays the 6pm-8pm Eastern range high and low  
from 8pm through 12pm Eastern.')
```

```
eveningRangeWidthInput = input.int(2, '6pm-8pm Level Width', minval = 1,  
maxval = 5, group = 'Display Settings', tooltip = 'Width of the 6pm-8pm range  
high and low lines.')
```

```
showRangeReactionLabelsInput = input.bool(true, 'Show Range  
Acceptance/Reversal', group = 'Display Settings', tooltip = 'Displays  
Acceptance and Reversal labels for the 6pm-8pm range when the FVG-123  
conditions qualify.')
```

```
showGoodLabelsInput = input.bool(true, 'Show Good Labels', group = 'Display  
Settings', tooltip = 'Displays only Good labels when a qualifying MFB CISC  
occurs.')
```

```
levelLineWidthInput = input.int(2, 'Key Level Width', minval = 1, maxval = 5,  
group = 'Display Settings', tooltip = 'Width of the plotted key-level lines.')
```

```
// --- Inputs: Colors ---
```

```
pdhColorInput = input.color(color.new(#ab47bc, 30), 'PDH/PDL Color', group
= 'Style Settings', inline = 'pdh', tooltip = 'Color used for previous-day high and
low.')
```

```
pwhColorInput = input.color(color.new(#5b9cf6, 30), 'PWH/PWL Color', group
= 'Style Settings', inline = 'pwh', tooltip = 'Color used for previous-week high
and low.')
```

```
sessionColorInput = input.color(color.new(color.gray, 50), 'Session Levels
Color', group = 'Style Settings', inline = 'session', tooltip = 'Color used for the
Asian, London, and New York opening-range levels.')
```

```
eveningRangeColorInput = input.color(#f23645, '6pm-8pm Levels Color',
group = 'Style Settings', inline = 'evening', tooltip = 'Color used for the 6pm-
8pm range high and low.')
```

```
eveningRangeMidColorInput = input.color(#fdd835, '6pm-8pm 50% Color',
group = 'Style Settings', tooltip = 'Color used for the 50% midpoint of the 6pm-
8pm range.')
```

```
bullGoodColorInput = input.color(#089981, 'Bullish Good Color', group =
'Style Settings', tooltip = 'Color used for bullish Good labels.')
```

```
bearGoodColorInput = input.color(#f23645, 'Bearish Good Color', group =
'Style Settings', tooltip = 'Color used for bearish Good labels.')
```

```
// --- Session and level functions ---
```

```
isInSession(string sessionInput) =>
```

```
    not na(time('D', sessionInput, NY_TIMEZONE))
```

```
isNewSessionBar(string sessionInput) =>
```

```
    int sessionTime = time('D', sessionInput, NY_TIMEZONE)
```

```
(na(sessionTime[1]) and not na(sessionTime)) or (not na(sessionTime[1]) and sessionTime[1] < sessionTime)
```

```
getSessionBounds(string sessionInput) =>
```

```
var float sessionHigh = na
```

```
var float sessionLow = na
```

```
bool newDay = ta.change(time('D', '0000-2359', NY_TIMEZONE)) != 0
```

```
if newDay
```

```
    sessionHigh := na
```

```
    sessionLow := na
```

```
if isInSession(sessionInput)
```

```
    bool newSessionBar = isNewSessionBar(sessionInput)
```

```
    sessionLow := newSessionBar ? low : na(sessionLow) ? low :
```

```
math.min(sessionLow, low)
```

```
    sessionHigh := newSessionBar ? high : na(sessionHigh) ? high :
```

```
math.max(sessionHigh, high)
```

```
[sessionHigh, sessionLow]
```

```
getEveningRangeBounds() =>
```

```
var float rangeHigh = na
```

```
var float rangeLow = na
```

```
bool inRange = not na(time('D', '1800-2000', NY_TIMEZONE))
```

```
bool newRangeBar = inRange and not inRange[1]
```

```
if newRangeBar
```

```

rangeHigh := high
rangeLow := low
else if inRange
    rangeHigh := na(rangeHigh) ? high : math.max(rangeHigh, high)
    rangeLow := na(rangeLow) ? low : math.min(rangeLow, low)
bool inExtension = not na(time('D', '2000-1200', NY_TIMEZONE))
bool rangeVisible = inRange or inExtension
[rangeVisible ? rangeHigh : na, rangeVisible ? rangeLow : na, inExtension]

// --- Key levels ---
[asianHigh, asianLow] = getSessionBounds('1800-0000')
[eveningRangeHigh, eveningRangeLow, eveningRangeExtended] =
getSessionRangeBounds()
eveningRangeMid = (eveningRangeHigh + eveningRangeLow) / 2
eveningHigh = eveningRangeExtended ? eveningRangeHigh : na
eveningLow = eveningRangeExtended ? eveningRangeLow : na
[londonHigh, londonLow] = getSessionBounds('0000-0600')
[opening15High, opening15Low] = getSessionBounds('0930-0945')
[opening30High, opening30Low] = getSessionBounds('0930-1000')
[opening60High, opening60Low] = getSessionBounds('0930-1030')

previousDayHigh = request.security(syminfo.tickerid, 'D', high[1], lookahead =
barmerge.lookahead_on)

```

```
previousDayLow = request.security(syminfo.tickerid, 'D', low[1], lookahead =  
barmerge.lookahead_on)
```

```
previousWeekHigh = request.security(syminfo.tickerid, 'W', high[1], lookahead  
= barmerge.lookahead_on)
```

```
previousWeekLow = request.security(syminfo.tickerid, 'W', low[1], lookahead  
= barmerge.lookahead_on)
```

```
pdhPlot = ta.change(time('D')) != 0 ? na : previousDayHigh
```

```
pdlPlot = ta.change(time('D')) != 0 ? na : previousDayLow
```

```
pwhPlot = ta.change(time('W')) != 0 ? na : previousWeekHigh
```

```
pwlPlot = ta.change(time('W')) != 0 ? na : previousWeekLow
```

```
plot(showLevelsInput ? pdhPlot : na, 'PDH', color = pdhColorInput, style =  
plot.style_linebr, linewidth = levelLineWidthInput)
```

```
plot(showLevelsInput ? pdlPlot : na, 'PDL', color = pdhColorInput, style =  
plot.style_linebr, linewidth = levelLineWidthInput)
```

```
plot(showLevelsInput ? pwhPlot : na, 'PWH', color = pwhColorInput, style =  
plot.style_linebr, linewidth = levelLineWidthInput)
```

```
plot(showLevelsInput ? pwlPlot : na, 'PWL', color = pwhColorInput, style =  
plot.style_linebr, linewidth = levelLineWidthInput)
```

```
plot(showEveningRangeInput ? eveningHigh : na, '6pm-8pm High (to 12pm)',  
color = eveningRangeColorInput, style = plot.style_linebr, linewidth =  
eveningRangeWidthInput)
```

```
plot(showEveningRangeInput ? eveningLow : na, '6pm-8pm Low (to 12pm)',  
color = eveningRangeColorInput, style = plot.style_linebr, linewidth =  
eveningRangeWidthInput)
```

```
plot(showEveningRangeInput and eveningRangeExtended ? eveningRangeMid  
: na, '6pm-8pm 50% (to 12pm)', color = eveningRangeMidColorInput, style =  
plot.style_linebr, linewidth = 1, linestyle = plot.linestyle_dotted)
```

```
plot(showLevelsInput ? asianHigh : na, 'Asian High', color =  
sessionColorInput, style = plot.style_linebr, linewidth = 1)
```

```
plot(showLevelsInput ? asianLow : na, 'Asian Low', color = sessionColorInput,  
style = plot.style_linebr, linewidth = 1)
```

```
plot(showLevelsInput ? londonHigh : na, 'London High', color =  
sessionColorInput, style = plot.style_linebr, linewidth = 1)
```

```
plot(showLevelsInput ? londonLow : na, 'London Low', color =  
sessionColorInput, style = plot.style_linebr, linewidth = 1)
```

```
plot(showLevelsInput ? opening15High : na, '9:30 Open High', color =  
sessionColorInput, style = plot.style_linebr, linewidth = 1)
```

```
plot(showLevelsInput ? opening15Low : na, '9:30 Open Low', color =  
sessionColorInput, style = plot.style_linebr, linewidth = 1)
```

```
plot(showLevelsInput ? opening30High : na, '10:00 Open High', color =  
sessionColorInput, style = plot.style_linebr, linewidth = 1)
```

```
plot(showLevelsInput ? opening30Low : na, '10:00 Open Low', color =  
sessionColorInput, style = plot.style_linebr, linewidth = 1)
```

```
plot(showLevelsInput ? opening60High : na, '10:30 Open High', color =  
sessionColorInput, style = plot.style_linebr, linewidth = 1)
```

```
plot(showLevelsInput ? opening60Low : na, '10:30 Open Low', color =  
sessionColorInput, style = plot.style_linebr, linewidth = 1)
```

```
// --- Timeframe filters ---
```

```
is1mChart = timeframe.isintraday and timeframe.multiplier == 1 and  
timeframe.isminutes
```

```
is3mChart = timeframe.isintraday and timeframe.multiplier == 3 and  
timeframe.isminutes
```

```
is5mChart = timeframe.isintraday and timeframe.multiplier == 5 and  
timeframe.isminutes
```

```
isSupportedChart = timeframe.isintraday and (timeframe.multiplier == 1 or  
timeframe.multiplier == 3 or timeframe.multiplier == 5 or timeframe.multiplier  
== 15 or timeframe.multiplier == 30 or timeframe.multiplier == 60) and  
timeframe.isminutes
```

```
// --- Key-level touch logic from the MFB source ---
```

```
highLevelTouch = high >= previousDayHigh or high >= eveningHigh or high >=  
asianHigh or high >= londonHigh or high >= opening15High or high >=  
opening30High or high >= opening60High or high >= previousWeekHigh
```

```
lowLevelTouch = low <= previousDayLow or low <= eveningLow or low <=  
asianLow or low <= londonLow or low <= opening15Low or low <=  
opening30Low or low <= opening60Low or low <= previousWeekLow
```

```
lastHighTouch = ta.barssince(highLevelTouch)
```

```
lastLowTouch = ta.barssince(lowLevelTouch)
```

```
// --- Displacement model ---
```

```
atrValue = ta.atr(atrLengthInput)
```

```
trueRangeValue = ta.tr(true)
```

```
isGoodApproach(bool bullishApproach) =>
```

```
    int validCount = 0
```

```

int qualifyingCount = 0

for i = 0 to approachLookbackInput - 1

    bool validBar = not na(atrValue[i]) and not na(trueRangeValue[i]) and not
na(close[i])

    if validBar

        validCount += 1

        float candleRange = math.max(high[i] - low[i], syminfo.mintick)

        float displacement = trueRangeValue[i] / atrValue[i]

        float closureEfficiency = bullishApproach ? (close[i] - low[i]) /
candleRange : (high[i] - close[i]) / candleRange

        bool directionallyAligned = bullishApproach ? close[i] > open[i] : close[i]
< open[i]

        bool qualifies = directionallyAligned and displacement >=
displacementThresholdInput and closureEfficiency >=
minClosureEfficiencyInput

        if qualifies

            qualifyingCount += 1

    validCount == approachLookbackInput and qualifyingCount >=
minQualifyingCandlesInput

// A high is approached bullishly; a low is approached bearishly.
bullApproachGood = isGoodApproach(false)
bearApproachGood = isGoodApproach(true)

var bool touchedHigh = false

```

```
var bool touchedLow = false
```

```
var bool highApproachGood = false
```

```
var bool lowApproachGood = false
```

```
if highLevelTouch
```

```
    touchedHigh := true
```

```
    highApproachGood := bullApproachGood
```

```
if lowLevelTouch
```

```
    touchedLow := true
```

```
    lowApproachGood := bearApproachGood
```

```
if not na(lastHighTouch) and lastHighTouch > primingLookbackInput
```

```
    touchedHigh := false
```

```
    highApproachGood := false
```

```
if not na(lastLowTouch) and lastLowTouch > primingLookbackInput
```

```
    touchedLow := false
```

```
    lowApproachGood := false
```

```
// --- CHoCH and CISD rules from the MFB source ---
```

```
var float lastSwingHigh = na
```

```
var float lastSwingLow = na
```

```
pivotHigh = ta.pivohigh(high, 3, 3)
```

```
pivotLow = ta.pivotlow(low, 3, 3)
```

```
if not na(pivotHigh)
```

```
    lastSwingHigh := pivotHigh
```

```
if not na(pivotLow)
```

```
    lastSwingLow := pivotLow
```

```
var float lastBearFvgTop = na
```

```
var float lastBullFvgBottom = na
```

```
bearishFvg123 = high < low[2]
```

```
bullishFvg123 = low > high[2]
```

```
if bearishFvg123
```

```
    lastBearFvgTop := low[2]
```

```
if bullishFvg123
```

```
    lastBullFvgBottom := high[2]
```

```
chochBullEvent = ta.crossover(close, lastSwingHigh)
```

```
chochBearEvent = ta.crossunder(close, lastSwingLow)
```

```
cisdBullEvent = ta.crossover(close, lastBearFvgTop)
```

```
cisdBearEvent = ta.crossunder(close, lastBullFvgBottom)
```

```
// --- 6pm-8pm range Acceptance and Reversal rules ---
```

```
eveningRangeInSession = not na(time('D', '1800-2000', NY_TIMEZONE))
```

```
newEveningRange = eveningRangeInSession and not  
eveningRangeInSession[1]
```

```
var bool eveningHighBroken = false
var bool eveningLowBroken = false
var bool eveningHighMidSwept = false
var bool eveningLowMidSwept = false
var bool eveningHighReactionDone = false
var bool eveningLowReactionDone = false
var int eveningHighBreakBar = na
var int eveningLowBreakBar = na
```

```
if newEveningRange
    eveningHighBroken := false
    eveningLowBroken := false
    eveningHighMidSwept := false
    eveningLowMidSwept := false
    eveningHighReactionDone := false
    eveningLowReactionDone := false
    eveningHighBreakBar := na
    eveningLowBreakBar := na
```

```
eveningHighCross = ta.crossover(close, eveningHigh)
eveningLowCross = ta.crossunder(close, eveningLow)
```

eveningHighBreakEvent = barstate.isconfirmed and eveningRangeExtended
and eveningHighCross

eveningLowBreakEvent = barstate.isconfirmed and eveningRangeExtended
and eveningLowCross

if eveningHighBreakEvent

 eveningHighBroken := true

 eveningHighBreakBar := bar_index

if eveningLowBreakEvent

 eveningLowBroken := true

 eveningLowBreakBar := bar_index

bool afterEveningHighBreak = eveningHighBroken and not
na(eveningHighBreakBar) and bar_index > eveningHighBreakBar

bool afterEveningLowBreak = eveningLowBroken and not
na(eveningLowBreakBar) and bar_index > eveningLowBreakBar

if afterEveningHighBreak and not na(eveningRangeMid) and low <=
eveningRangeMid

 eveningHighMidSwept := true

if afterEveningLowBreak and not na(eveningRangeMid) and high >=
eveningRangeMid

 eveningLowMidSwept := true

bool eveningHighAcceptance = afterEveningHighBreak and not
eveningHighReactionDone and bullishFvg123 and close > eveningHigh

bool eveningLowAcceptance = afterEveningLowBreak and not
eveningLowReactionDone and bearishFvg123 and close < eveningLow

bool eveningHighReversal = afterEveningHighBreak and not
eveningHighReactionDone and eveningHighMidSwept and bearishFvg123

bool eveningLowReversal = afterEveningLowBreak and not
eveningLowReactionDone and eveningLowMidSwept and bullishFvg123

if eveningHighAcceptance or eveningHighReversal

 eveningHighReactionDone := true

if eveningLowAcceptance or eveningLowReversal

 eveningLowReactionDone := true

bullChd = (is1mChart or is3mChart) and touchedLow and (chochBullEvent or
cisdBullEvent) and close > lastSwingHigh and close > lastBearFvgTop

bearChd = (is1mChart or is3mChart) and touchedHigh and (chochBearEvent
or cisdBearEvent) and close < lastSwingLow and close < lastBullFvgBottom

// --- 5m Cisd rules from the MFB source ---

var float displacedHighLevel = na

var float displacedHighLow = na

var bool isHighDisplaced = false

var int displacedHighBar = na

var float displacedLowLevel = na

var float displacedLowHigh = na

var bool isLowDisplaced = false

var int displacedLowBar = na

crossAbovePreviousDayHigh = ta.crossover(close, previousDayHigh)

crossAbovePreviousWeekHigh = ta.crossover(close, previousWeekHigh)

crossAboveEveningHigh = ta.crossover(close, eveningHigh)

crossAboveAsianHigh = ta.crossover(close, asianHigh)

crossAboveLondonHigh = ta.crossover(close, londonHigh)

crossAboveOpening15High = ta.crossover(close, opening15High)

crossAboveOpening30High = ta.crossover(close, opening30High)

crossAboveOpening60High = ta.crossover(close, opening60High)

crossBelowPreviousDayLow = ta.crossunder(close, previousDayLow)

crossBelowPreviousWeekLow = ta.crossunder(close, previousWeekLow)

crossBelowEveningLow = ta.crossunder(close, eveningLow)

crossBelowAsianLow = ta.crossunder(close, asianLow)

crossBelowLondonLow = ta.crossunder(close, londonLow)

crossBelowOpening15Low = ta.crossunder(close, opening15Low)

crossBelowOpening30Low = ta.crossunder(close, opening30Low)

crossBelowOpening60Low = ta.crossunder(close, opening60Low)

bullishBreach = crossAbovePreviousDayHigh or crossAboveEveningHigh or
crossAboveAsianHigh or crossAboveLondonHigh or
crossAboveOpening15High or crossAbovePreviousWeekHigh or
crossAboveOpening30High or crossAboveOpening60High

bearishBreach = crossBelowPreviousDayLow or crossBelowEveningLow or
crossBelowAsianLow or crossBelowLondonLow or crossBelowOpening15Low
or crossBelowPreviousWeekLow or crossBelowOpening30Low or
crossBelowOpening60Low

float breachedHighLevel = na

if crossAbovePreviousDayHigh

breachedHighLevel := previousDayHigh

if crossAbovePreviousWeekHigh

breachedHighLevel := na(breachedHighLevel) ? previousWeekHigh :
math.max(breachedHighLevel, previousWeekHigh)

if crossAboveEveningHigh

breachedHighLevel := na(breachedHighLevel) ? eveningHigh :
math.max(breachedHighLevel, eveningHigh)

if crossAboveAsianHigh

breachedHighLevel := na(breachedHighLevel) ? asianHigh :
math.max(breachedHighLevel, asianHigh)

if crossAboveLondonHigh

breachedHighLevel := na(breachedHighLevel) ? londonHigh :
math.max(breachedHighLevel, londonHigh)

if crossAboveOpening15High

```
breachedHighLevel := na(breachedHighLevel) ? opening15High :  
math.max(breachedHighLevel, opening15High)
```

```
if crossAboveOpening30High
```

```
breachedHighLevel := na(breachedHighLevel) ? opening30High :  
math.max(breachedHighLevel, opening30High)
```

```
if crossAboveOpening60High
```

```
breachedHighLevel := na(breachedHighLevel) ? opening60High :  
math.max(breachedHighLevel, opening60High)
```

```
float breachedLowLevel = na
```

```
if crossBelowPreviousDayLow
```

```
breachedLowLevel := previousDayLow
```

```
if crossBelowPreviousWeekLow
```

```
breachedLowLevel := na(breachedLowLevel) ? previousWeekLow :  
math.min(breachedLowLevel, previousWeekLow)
```

```
if crossBelowEveningLow
```

```
breachedLowLevel := na(breachedLowLevel) ? eveningLow :  
math.min(breachedLowLevel, eveningLow)
```

```
if crossBelowAsianLow
```

```
breachedLowLevel := na(breachedLowLevel) ? asianLow :  
math.min(breachedLowLevel, asianLow)
```

```
if crossBelowLondonLow
```

```
breachedLowLevel := na(breachedLowLevel) ? londonLow :  
math.min(breachedLowLevel, londonLow)
```

```
if crossBelowOpening15Low
```

```
breachedLowLevel := na(breachedLowLevel) ? opening15Low :  
math.min(breachedLowLevel, opening15Low)
```

```
if crossBelowOpening30Low
```

```
breachedLowLevel := na(breachedLowLevel) ? opening30Low :  
math.min(breachedLowLevel, opening30Low)
```

```
if crossBelowOpening60Low
```

```
breachedLowLevel := na(breachedLowLevel) ? opening60Low :  
math.min(breachedLowLevel, opening60Low)
```

```
if is5mChart
```

```
if bullishBreach
```

```
isHighDisplaced := true
```

```
displacedHighLevel := breachedHighLevel
```

```
displacedHighLow := low
```

```
displacedHighBar := bar_index
```

```
highApproachGood := bullApproachGood
```

```
if bearishBreach
```

```
isLowDisplaced := true
```

```
displacedLowLevel := breachedLowLevel
```

```
displacedLowHigh := high
```

```
displacedLowBar := bar_index
```

```
lowApproachGood := bearApproachGood
```

```
if is5mChart
```

```
if isHighDisplaced and close > displacedHighLevel and low >
displacedHighLow and bar_index - displacedHighBar > 10
```

```
isHighDisplaced := false
```

```
highApproachGood := false
```

```
if isLowDisplaced and close < displacedLowLevel and high <
displacedLowHigh and bar_index - displacedLowBar > 10
```

```
isLowDisplaced := false
```

```
lowApproachGood := false
```

```
bearFiveMinuteCisd = is5mChart and isHighDisplaced and close <
displacedHighLevel and close < displacedHighLow
```

```
bullFiveMinuteCisd = is5mChart and isLowDisplaced and close >
displacedLowLevel and close > displacedLowHigh
```

```
// Signals are confirmed only after candle close to avoid intrabar label
changes.
```

```
bullGoodSignal = barstate.isconfirmed and ((bullChd and lowApproachGood)
or (bullFiveMinuteCisd and lowApproachGood))
```

```
bearGoodSignal = barstate.isconfirmed and ((bearChd and
highApproachGood) or (bearFiveMinuteCisd and highApproachGood))
```

```
plotshape(showRangeReactionLabelsInput and eveningHighAcceptance, title
= '6pm-8pm Bullish Acceptance', text = 'Acceptance', style = shape.labelup,
location = location.belowbar, color = #00000000, textcolor =
eveningRangeMidColorInput, size = size.small)
```

```
plotshape(showRangeReactionLabelsInput and eveningLowAcceptance, title = '6pm-8pm Bearish Acceptance', text = 'Acceptance', style = shape.labeldown, location = location.abovebar, color = #00000000, textcolor = eveningRangeMidColorInput, size = size.small)
```

```
plotshape(showRangeReactionLabelsInput and eveningHighReversal, title = '6pm-8pm Bearish Reversal', text = 'Reversal', style = shape.labeldown, location = location.abovebar, color = #00000000, textcolor = eveningRangeMidColorInput, size = size.small)
```

```
plotshape(showRangeReactionLabelsInput and eveningLowReversal, title = '6pm-8pm Bullish Reversal', text = 'Reversal', style = shape.labelup, location = location.belowbar, color = #00000000, textcolor = eveningRangeMidColorInput, size = size.small)
```

```
plotshape(showGoodLabelsInput and bullGoodSignal, title = 'Bullish Good', text = 'Good', style = shape.labelup, location = location.belowbar, color = bullGoodColorInput, textcolor = color.white, size = size.small)
```

```
plotshape(showGoodLabelsInput and bearGoodSignal, title = 'Bearish Good', text = 'Good', style = shape.labeldown, location = location.abovebar, color = bearGoodColorInput, textcolor = color.white, size = size.small)
```

```
if bullFiveMinuteCisd
```

```
    isLowDisplaced := false
```

```
if bearFiveMinuteCisd
```

```
    isHighDisplaced := false
```

```
if bullChd
```

```
    touchedLow := false
```

```
    lowApproachGood := false
```

```
if bearChd
```

touchedHigh := false

highApproachGood := false

if bullGoodSignal

 alert('MFB - Trade Entry Quality: Good bullish CISD on ' + syminfo.ticker,
 alert.freq_once_per_bar_close)

if bearGoodSignal

 alert('MFB - Trade Entry Quality: Good bearish CISD on ' + syminfo.ticker,
 alert.freq_once_per_bar_close)

alertcondition(bullGoodSignal, 'Good Bullish CISD', 'MFB - Trade Entry Quality:
Good bullish CISD on {{ticker}}')

alertcondition(bearGoodSignal, 'Good Bearish CISD', 'MFB - Trade Entry
Quality: Good bearish CISD on {{ticker}}')

if not isSupportedChart and barstate.islast

 runtime.error(DESIGNATED_TIMEFRAMES)