```vb
Option Explicit

Const ReturnNoSolution As Integer = -32767
Const ReturnNotMovableDelta As Integer = -1000

Dim RecurLevel As Integer
Dim ListRecurCall As String



'-------------------------------------------------------------
AU_NPM_MarginManageProtectedFlights
' modify the FDATime
' lprioModel must be assigned to GPrioModel_TimeMode_OnSchedule or
GPrioModel_TimeMode_OnMargin
'  in this case we use the schedule or the margin to make the calculation

' lAllFl  is list off flights to find Margin solution (All except Pflights and ExplicitB
flights)
' when finding margin solution, if default priority value is baseline, prio is put to lowest
prio
' If  Time not after is greater then the Hotspot end, initialize to the end of the hotspot
' BE CAREFULL a B priority on margins flights exclude flight from management
Function AU_NPM_ManageMarginPrioFlights_Main(lprioModel As Integer, _
            lAll_AUFlights As CL_AllFlights, lMyFlightsIx As CL_AUFlightsIx, _
            ByRef lAllFl() As Integer, lAllFl_nb As Integer, _
            ByRef lAddFlight() As Integer, ByRef lMarginflight_Nb As Integer) As Integer

    Dim li As Integer
    Dim lj As Integer
    Dim lName As String
    Dim ltime As Date


    'Dim lAllFl_prio() As Integer ' containt prio off All flight for this AU (Ix in
Allflight)

    'Dim lAllFlSortedOnBaseline() As Integer ' containt All flight to manage the margin
sorted

    Dim lMarginFl() As Integer ' containt All flight twho have a  margin
    Dim lMarginSortedFl() As Integer ' containt All flight to manage the margin sorted
    Dim lMarginFl_nb As Integer ' nb of flight to manage the margin

    Dim lPrioOnlyFl() As Integer ' containt All flight twho have a  margin
    Dim lPrioOnlySortedFl() As Integer ' containt All flight to manage the margin sorted
    Dim lPrioOnlyFl_nb As Integer ' nb of flight to manage the margin



    Call EX_Mess(EX_MESS_Start, "AU NPRIO MARGINandPrio: " & lAll_AUFlights.AUName)
    'ListRecurCall = ""

    ' do nothing if only 1 flight
    If lAllFl_nb < 2 Then
        Exit Function
    End If


    Dim lMargeIx As Integer

    ' slot management
    Dim lMySlotsValue() As Date
    Dim lMySlotsValueSorted() As Date
    Dim lMySlotsUsed() As Integer

    ' get the flight with margin to manage
    ' get the list of flight with margins to manage


    lMarginFl_nb = 0
    lPrioOnlyFl_nb = 0

    ' get margin flights and prio only flights
    ' BE CAREFULL the management of explicit B on margin flight
    ' explicit B normally not part of the flights
```



```vb
    If lAllFl_nb > 0 Then
        ReDim lMarginFl(lAllFl_nb)
        ReDim lMySlotsValue(lAllFl_nb)
        ReDim lPrioOnlyFl(lAllFl_nb)
        ReDim lMySlotsUsed(lAllFl_nb)
        For li = 0 To lAllFl_nb - 1
            lMySlotsValue(li) = lAll_AUFlights.GetFDATime(lAllFl(li))
            lMySlotsUsed(li) = -1
            If (lAll_AUFlights.GetMarginNotAfterTimeIsInit(lAllFl(li)) = True) And _
                (lAll_AUFlights.GetPrio(lAllFl(li)) <> GPrioSuspended) Then
                lMarginFl(lMarginFl_nb) = lAllFl(li)
                lMarginFl_nb = lMarginFl_nb + 1
            Else
                lPrioOnlyFl(lPrioOnlyFl_nb) = lAllFl(li)
                lPrioOnlyFl_nb = lPrioOnlyFl_nb + 1
            End If
        Next li

        ' get my slots and manage the list of margin or prio flights
        ReDim lMySlotsValueSorted(lAllFl_nb)

        ' Sort my Time slots (by FDATime)
        Call AU_NPS_SortATimeTable(lMySlotsValue, lMySlotsValueSorted, lAllFl_nb)
        Erase lMySlotsValue


        If lMarginFl_nb < 1 Then
            Erase lMarginFl
        Else

            ' Sort my Margin flights by prio and Margins and Schedule
            ReDim lMarginSortedFl(lMarginFl_nb)
            Call AU_NPS_SortByPrioAndMarginTimeNotAfterAndBaselineTime(lAll_AUFlights,
lMarginFl, lMarginSortedFl, lMarginFl_nb)
            Erase lMarginFl

            ' loop on earch Margin flights
            ' put Margin flights on available slot

                Call EX_Log_Init

            For lMargeIx = 0 To lMarginFl_nb - 1

                ' for test
                Dim lCallsign As String
                Dim lMarginTime As Date
                Dim lMarginFlightIx As Integer

                lMarginFlightIx = lMarginSortedFl(lMargeIx)
                lCallsign = lAll_AUFlights.GetCallsignICAO(lMarginFlightIx)

                Dim lSlotAssigned As Integer

                If lMargeIx = 73 Then
                    lMargeIx = lMargeIx
                End If

                lMarginTime = lAll_AUFlights.GetMarginNotAfterTime(lMarginFlightIx)

                lSlotAssigned =
AU_NPM_ManageMarginPrioFlights_ManageTimeSolution(lAll_AUFlights, _
                    lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
                    lMarginTime, lMarginFlightIx)



                ' Test if there is a slot
                If lSlotAssigned < 0 Then
                    ' no time solution
                    Call AU_NPM_MsgboxStop(" ERROR TO ASSIGN Margin flight to a slot : " &
lCallsign, _
                            lAll_AUFlights, lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb)
                End If

            Next lMargeIx
        End If

```

```
        ' manage the other type of flights
        If lPrioOnlyFl_nb < 1 Then
            'Erase lMySlotsValue
            Erase lPrioOnlyFl
            'Erase lMySlotsUsed
        Else
            ' AU slot has been assigned to Margin flights
            ' now manage the other flights (prio only)
            ReDim lPrioOnlySortedFl(lPrioOnlyFl_nb)
            Call AU_NPS_SortByPrioAndSchedule(lAll_AUFlights, lPrioOnlyFl,
lPrioOnlySortedFl, lPrioOnlyFl_nb)
            Erase lPrioOnlyFl

            ' assign prio flights be carrefull baseline flights must be assign first then
flight with number
            Call AU_NPM_ManageMarginPrioFlights_AssignOtherFlights(lAll_AUFlights, _
                lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
                lPrioOnlySortedFl(), lPrioOnlyFl_nb)

            Erase lMarginSortedFl

        End If


        ' return the nb of margin flights
        lMarginflight_Nb = lMarginFl_nb
    End If

    ' at the en Pack the flights on slots (use the available AU slot
    ' needed if there is some Suspended flights
    If lMarginFl_nb > 0 Or lPrioOnlyFl_nb > 0 Then
        Call AU_NPM_ManageMarginPrioFlights_UseAvailableSlots(lAll_AUFlights, _
                lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb)

    End If

    ' assign the flights to the slots
    '------ Update the FDA time ------
    ' First assigne the FDA time on margins flights
    For li = 0 To lAllFl_nb - 1
        ltime = lMySlotsValueSorted(li)
        ' if the slot is used by a margin flight assign it
        If lMySlotsUsed(li) <> -1 Then
            ' assign the Margin flights
            lName = lAll_AUFlights.GetCallsignICAO(lMySlotsUsed(li))
            Call lAll_AUFlights.SetFDATime(lMySlotsUsed(li), lMySlotsValueSorted(li))
        End If
    Next li

    ' ---------- Manage the Suspended flights
    If lPrioOnlyFl_nb > 0 Then

        ' Assign it the hotspot end time in fdaValue
        Call AU_NPM_ManageMarginPrioFlights_AssignSuspendedFlights(lAll_AUFlights, _
                lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
                lPrioOnlySortedFl(), lPrioOnlyFl_nb)
    End If

    Erase lPrioOnlySortedFl
    Erase lMySlotsValueSorted
    Erase lMySlotsUsed


    ' return nb of impacted flights to add on next list
    AU_NPM_ManageMarginPrioFlights_Main = 0

    Call EX_Mess(EX_MESS_End, "AU NPRIO MARGINandPrio: " & lAll_AUFlights.AUName)


End Function

'---------------------------------------- AU_NPM_ManageMarginPrioFlights_ManageTimeSolution
' try and manage the solution before the target time
' At this stage the target slot is already used by another flight
' first look if there is an available slot before the target one to shift earlier the others
' if yes :
'     try to shift the flights to make a hole at the target slot
'     otherwise give a slot before for this flight
' if no slot return -1
Function AU_NPM_ManageMarginPrioFlights_ManageTimeSolution( _
            lAll_AUFlights As CL_AllFlights, _
            ByRef lMySlotsValueSorted() As Date, ByRef lMySlotsUsed() As Integer, lAllFl_nb
As Integer, _
            lTargetTime As Date, _
            lFlightIx As Integer) As Integer

    'Dim lSlotEarlierPossible As Integer

    Dim lAvailableSlot_Earlier As Integer
    Dim lAvailableSlot_Later As Integer

    Dim lEarliestTime As Date
    Dim lTargetSlot As Integer

    Dim lReturn As Integer

    'lAvailableSlot_Earlier = -1

    RecurLevel = 0
    'Call EX_Log_Init

    ' earliest time of the margin flight
    lEarliestTime = lAll_AUFlights.GetRefBlockTime(lFlightIx) _
                                                - GHspt_FlightEarlyDeparture_forDate



    lTargetSlot = AU_NPMF_GetTargetSlots(lMySlotsValueSorted, lAllFl_nb, lTargetTime,
lEarliestTime)

    ' Test if there is a slot
    If lTargetSlot < 0 Then
        ' no time solution
        Call AU_NPM_MsgboxStop(" ERROR TO DO SOMETHING Margin flight with no time solution :
" & lAll_AUFlights.GetCallsignICAO(lFlightIx), _
                lAll_AUFlights, lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb)

                Call EX_Log(RecurLevel, "  --> No slot solution for " & lFlightIx & " on
targetTime " & lTargetTime)
        lReturn = -1
    Else
        ' try manage solution by shifting flights earlier first
        ' the slot could ber later then the one asked
        lAvailableSlot_Earlier =
AU_NPM_ManageMarginPrioFlights_ManageSolutionEarlier(lAll_AUFlights, _
                lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
                lTargetSlot, lFlightIx)

        If lAvailableSlot_Earlier > -1 Then
            ' Found a place in the slot list earlierassign it
            lMySlotsUsed(lAvailableSlot_Earlier) = lFlightIx
            'ListRecurCall = ListRecurCall & " S= " & lAvailableSlot_Earlier & "   |"
            Call EX_Log(RecurLevel, "End FL: " & lFlightIx & " EARLIER Solution is slot: " &
lAvailableSlot_Earlier)


            lReturn = lAvailableSlot_Earlier
        Else
            ' No possible slot earlier
            ' test and get if there is available slot later
            lAvailableSlot_Later = AU_NPMF_GetLaterAvailableSlots(lMySlotsValueSorted(),
lMySlotsUsed(), lAllFl_nb, _
                                        lTargetSlot + 1, lEarliestTime)

            If lAvailableSlot_Later > -1 Then
                ' put it at this place
                lMySlotsUsed(lAvailableSlot_Later) = lFlightIx
                Call EX_Log(RecurLevel, "End FL: " & lFlightIx & " LATER Solution is slot: "
& lAvailableSlot_Later)
                lReturn = lAvailableSlot_Later
            Else
                ' in this case no possibility to shift flight earlier
                ' and no slot available after
                ' otherwhise there is other possibilities .......
```

```
                lReturn = -1
                Call EX_Log(RecurLevel, " --> ERR: No slot for " & lFlightIx)

                Call AU_NPM_MsgboxStop("ManageTimeSolution PB of NB of available slot not OK for : " & _
                        lAll_AUFlights.GetCallsignICAO(lFlightIx) & " id= " & lFlightIx, _
                        lAll_AUFlights, lMySlotsValueSorted, lMySlotsUsed, lAllFl_nb)
            End If
        End If
    End If


    AU_NPM_ManageMarginPrioFlights_ManageTimeSolution = lReturn

End Function



'----------------------------------------
AU_NPM_ManageMarginPrioFlights_ManageSolutionEarlier
' Manage the solution at or before the target time or later if not found ealier
' At this stage the target slot is already used by another flight
'      try to manage the solution by shifting the flight who occupy the slot earlier
'      because this flight has a higher priority because it's assigned before

'Input :
'    - the list of slot
'    - the slot used in this list by a previous assigned flight,
'    - the flight to be managed
'    - the started target slot

'Output:
'    - the available slot for this flight


' first check if there is an available slot before the target one (needed to find a solution earlier)
' if No available slot before : stop this function and return -1
' if there is an empty slot before (minimum constraint to have a earlier solution)
'    loop from the current needed slot to the latest slot of the list
'                        (look also on later slot to return a slot later if no solution before)
'         Call "MoveFlightEarlier" the recursive function to try to shift the flights on earlier slot
'                    to make a hole to assign the flight on a slot
'         if a slot is found: return the slot found (end of loop)
'     End of the loop (at this stage the slot found could be later)

' if no possible slot return -1
' otherwise return the slot found


Function AU_NPM_ManageMarginPrioFlights_ManageSolutionEarlier( _
            lAll_AUFlights As CL_AllFlights, _
            ByRef lMySlotsValueSorted() As Date, ByRef lMySlotsUsed() As Integer, lAllFl_nb As Integer, _
            lTargetSlot As Integer, _
            lMarginFlightIx As Integer) As Integer

    Dim lSlotEarlierPossible As Integer
    Dim lAvailableSlot_Earlier As Integer
    Dim lTargetMove As Integer

    lAvailableSlot_Earlier = -1

' to test
If lAll_AUFlights.GetCallsignICAO(lMarginFlightIx) = "AFR165Z" Then
    lMarginFlightIx = lMarginFlightIx
End If


    lSlotEarlierPossible = AU_NPMF_GetIxOfEarlierFlightCanMove(lAll_AUFlights, _
                lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, lTargetSlot, lMarginFlightIx)

    ' some time say no but there is SO overwrite it
    'lSlotEarlierPossible = 0
```



```

    If lSlotEarlierPossible > -1 Then
    ' move flight earlier is possible
    'ListRecurCall = ListRecurCall & " | T:" & lMarginFlightIx & " " & lTargetSlot & " -> "


        ' loop until solution found or end of slots
        ' apply the recursive move flight earlier function from the target slot to the end of the slots
        ' until a solution is found
        For lTargetMove = lTargetSlot To lAllFl_nb - 1
            'ListRecurCall = "FL: " & lMarginFlightIx & " Slot: " & lTargetMove & " -> "
            Call EX_Log(RecurLevel, "Start FL: " & lMarginFlightIx & " Slot: " & lTargetMove & " -> ")

            ' test and manage if there is a slot earlier
            lAvailableSlot_Earlier = AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier(lAll_AUFlights, _
                lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
                lTargetMove, lMarginFlightIx)

            'ListRecurCall = ListRecurCall & " Target= " & lTargetMove & " Out=" & lAvailableSlot_Earlier")

            If lAvailableSlot_Earlier > -1 Then
                ' Found a place in the slot list earlierassign it
                'lMySlotsUsed(lAvailableSlot_Earlier) = lMarginFlightIx
                lTargetMove = lAllFl_nb
                'ListRecurCall = ListRecurCall & " S= " & lAvailableSlot_Earlier & "   |"
                Call EX_Log(RecurLevel, "En FL: " & lMarginFlightIx & " Slot: " & lAvailableSlot_Earlier)

            Else
                'ListRecurCall = ListRecurCall & " ..Next.."
                Call EX_Log(RecurLevel, "   FL: " & lMarginFlightIx & " Slot Not OK .. Next")
            End If

        Next lTargetMove



        If lAvailableSlot_Earlier < 0 Then

            ' LG2018-07 No now it's OK because use of all earlier slots
            ' If a flight can't be on the earlier slot it's meen that ther is a slot later !!!
            ' SO only return -1

            Call EX_Log(RecurLevel, "En FL: " & lMarginFlightIx & " No Solution found earlier even with hole")
GoTo lNext1:
            ' a solution can exist
            ' here it's because an affected flight lock the list because it's after another one

            If AU_NPM_ManageMarginPrioFlights_IsFlightScheduleCompatible(lAll_AUFlights, _
                    lMySlotsValueSorted(lSlotEarlierPossible), lMarginFlightIx) = True Then
                ' this fligh is compatible
                lAvailableSlot_Earlier = lSlotEarlierPossible
                Call EX_Log(RecurLevel, "En FL: " & lMarginFlightIx & " Direct Slot: " & lAvailableSlot_Earlier)
            Else
                ' no solution because current and earlier slot not compatible with schedule
                ' something wrong here
                Call EX_Log(RecurLevel, "En FL: " & lMarginFlightIx & " ERROR No slot ")

                Call AU_NPM_MsgboxStop("Margins PB possible slot before but not found solution !!! : " & _
                        lAll_AUFlights.GetCallsignICAO(lMarginFlightIx) & " id= " & lMarginFlightIx & vbCr & ListRecurCall, _
                        lAll_AUFlights, lMySlotsValueSorted, lMySlotsUsed, lAllFl_nb)
            End If

lNext1:

        End If
```

```
    Else
         lAvailableSlot_Earlier = -1
    End If


    AU_NPM_ManageMarginPrioFlights_ManageSolutionEarlier = lAvailableSlot_Earlier

End Function




' Recursive function to find and return a slot for a flight
' If the slot is not free it make it available by shifting already assign flights earlier (the already assign flights have higher priority)
' this function make the shift of the flights only if all flights can be shifted (recursive test before shifting)
'Input :
'     - the list of slot
'     - the slot used in this list by a previous assigned flight,
'     - the flight to be managed
'     - the started target slot

'Output:
'     - return >-1 : the available slot position set to free for this flight
'     - return = -1  blocking point, no possible shift earlier
'     - return =  a negative value starting at -1000
'                          + the negative value of the slot corresponding to a Unmovable flight
'                           ex: -1051 : the slot 51 is occupied by a Unmovable flight



Function AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier(lAll_AUFlights As CL_AllFlights, _
            ByRef lMySlotsValueSorted() As Date, ByRef lMySlotsUsed() As Integer, lSlots_nb As Integer, _
            lTargetIx As Integer, lFlightIx_ToMove As Integer) As Integer

    Dim li As Integer
    Dim lReturn As Integer
    Dim lTo As Integer
    Dim lFrom As Integer
    Dim lTryPreviousFlight As Boolean
    Dim lEarliestTime As Date
    Dim lPos As Integer
    Dim lCallsign As String

    ' to test
    RecurLevel = RecurLevel + 1
    lCallsign = lAll_AUFlights.GetCallsignICAO(lFlightIx_ToMove)

    Call EX_Log(RecurLevel, "Mv FL: " & lFlightIx_ToMove & " to Slot: " & lTargetIx & "(used by " & lMySlotsUsed(lTargetIx) & ")")

' ca boucle ici !!!!!!!
If lFlightIx_ToMove = 284 And lTargetIx = 57 And lMySlotsUsed(lTargetIx) = 272 Then
    lTargetIx = lTargetIx
End If

   ' try to make the target slot available by shifting it earlier
    lTryPreviousFlight = True
    lFrom = lTargetIx
    lTo = lTargetIx - 1

   ' initial condition
   '     FromSlot = Target slot
   '     ToSlot = Target slot - 1  slot to be use for the first shift in case the target one is used

   ' loop on the flights until a solution is found or no possible solution
    While lTryPreviousFlight = True
        ' test if the flight to put on the target slot (FromSlot)is time compatible
        '      (with its reference time - Airport early schedule duration)
        If AU_NPM_ManageMarginPrioFlights_IsFlightScheduleCompatible(lAll_AUFlights, _
                lMySlotsValueSorted(lFrom), lFlightIx_ToMove) = False Then
```



```
        'if the flight is not time compatible with the FromSLot
            'stop the loop and return the index of the tested slot as negative value
            '          add -1000 to be sur the value 0 is well managed

                lReturn = ReturnNotMovableDelta - lFrom
                lTryPreviousFlight = False
                Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Slot: " & lFrom & " To Early .. Return : " & lReturn)
        Else
        'Else : the flight is compatible with the FromSlot
            If lMySlotsUsed(lFrom) = -1 Then
            'if the Fromslot is empty: no flight assigned on it
                'return this slot and stop the loop
                lReturn = lFrom
                lTryPreviousFlight = False
                Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Slot empty OK .. Return : " & lReturn)

            Else
            'Else : the slot is used
                'try to move the flight assigned in the used slot at an earlier position (to the ToSlot)
                'if the ToSlot is < 0 (it is not possible to move before because it was the first slot)
                '   in this case there is no solution:  and end the loop
                If lTo < 0 Then
                    ' we are at the beginning of the slot list without finding a solution
                    ' no possible solutions return -1
                    lReturn = ReturnNoSolution
                    lTryPreviousFlight = False
                    Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Slot -- No SOLUTION -- At the end of Slots list Return : " & lReturn)

                Else
                'Else: the slot is used
                    'try to move the flight using the ToSlot to a earlier position by using this same recursive function
                    'call lPos =  MoveflightEarlier with in parameter: with the ToSlot target and with the flight in the FromSlot position

                    lPos = AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier(lAll_AUFlights, _
                         lMySlotsValueSorted(), lMySlotsUsed(), lSlots_nb, _
                         lTo, lMySlotsUsed(lFrom))

                    'if the function return a positive value : (lPos > -1)

                    If lPos > -1 Then
                        'a compatible slot has been found
                        'make the use of this slot effective (assign it)
                            'put the flight use to call the recursive function on the Slot position returned by it
                            'empty the slot used by it previously
                            'end the loop and return the empty slot
                        lMySlotsUsed(lPos) = lMySlotsUsed(lFrom)
                        lMySlotsUsed(lFrom) = -1

                        lReturn = lFrom 'return the slot put to free by the shift
                        lTryPreviousFlight = False

                        Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Recur OK return " & lReturn & _
                                  " Flight move from: " & lFrom & " To: " & lPos)

                    ElseIf lPos = ReturnNoSolution Then
                    'Else if the function return a -1 value (no possible solution) : lPos = -1
                        ' return no possible solution and close the loop
                        ' no solution because flights impossible to move
                        ' cannot create a hole
                        lReturn = ReturnNoSolution
                        lTryPreviousFlight = False
                        Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Recur NO SOLUTION ")

                    Else
```

```
                    'Else : the function return a negative value < -1 a flight is blocked on its position lPos = -1xxx
                        'continue to loop with an earlier position

                        'FromSlot = Slot position used to find solution on a flight blocked :
                        '    Slot position -1 of the flight blocked
                        '   (be careful : corresponding to the ToSlot position when the function is called)
                        'ToSlot      = FromSlot - 1

                        '                       test if not the earliest possible Slot to test
                        '                        If FromSlot position is > -1 , continue the loop
                        '                        otherwise stop the loop and end by a -1 solution (no solution)
                        lFrom = -lPos + ReturnNotMovableDelta
                        lTo = lFrom - 1

                        If lFrom < 0 Then
                            ' From from next loop must be >= 0 otherwhise no solution, stop the loop
                            lTryPreviousFlight = False
                            lReturn = ReturnNoSolution

                            Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Recur NO solution no more Slot to check " & _
                                        lFrom)
                        Else
                            lTryPreviousFlight = True

                            Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Recur Next target Dde: " & _
                                        lTargetIx & " Check: " & lFrom)

                        End If

                    End If
                End If
            End If
        End If
    Wend

    AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier = lReturn
    RecurLevel = RecurLevel - 1
End Function









' Recursive function to find and return a slot for a flight
' If the slot is not free it make it available by shifting already assign flights earlier (the already assign flights have higher priority)
' this function make the shift of the flights only if all flights can be shifted (recursive test before shifting)
'Input :
'     - the list of slot
'     - the slot used in this list by a previous assigned flight,
'     - the flight to be managed
'     - the started target slot

'Output:
'     - return >-1 : the available slot position set to free for this flight
'     - return = -1  blocking point, no possible shift earlier
'     - return =  a negative value starting at -1000
'                         + the negative value of the slot corresponding to a Unmovable flight
'                          ex: -1051 : the slot 51 is occupied by a Unmovable flight


Function AU_NPM_ManageMarginPrioFlights_MoveFlightEarlierOLD(lAll_AUFlights As CL_AllFlights, _
```



```
            ByRef lMySlotsValueSorted() As Date, ByRef lMySlotsUsed() As Integer, lSlots_nb As Integer, _
            lTargetIx As Integer, lFlightIx_ToMove As Integer) As Integer

    Dim li As Integer
    Dim lReturn As Integer
    Dim lTo As Integer
    Dim lFrom As Integer
    Dim lTryPreviousFlight As Boolean
    Dim lEarliestTime As Date
    Dim lPos As Integer
    Dim lCallsign As String

    ' to test
    RecurLevel = RecurLevel + 1
    lCallsign = lAll_AUFlights.GetCallsignICAO(lFlightIx_ToMove)

    Call EX_Log(RecurLevel, "Rec Start Mv FL: " & lFlightIx_ToMove & " to Slot: " & lTargetIx & "(used by " & lMySlotsUsed(lTargetIx) & ")")


    ' make it available by shifting earlier previous flights
    lTryPreviousFlight = True
    lFrom = lTargetIx
    lTo = lTargetIx - 1


    ' loop on the flights because some of them could not be moved earlier due to schedule
    While lTryPreviousFlight = True
        ' test if the target slot is compatible (with reference time - Airport early schedule duration)
        If AU_NPM_ManageMarginPrioFlights_IsFlightScheduleCompatible(lAll_AUFlights, _
                lMySlotsValueSorted(lFrom), lFlightIx_ToMove) = False Then
                ' the flight is not compatible with the reference time of the target
                ' stop the loop on this flight and return the last tested flight (as negative value)
                ' add -1000 to be sur the value 0 is managed
                '  in recursive calling function, its the To target (in the current one it's the from or an earlier one)
                lReturn = ReturnNotMovableDelta - lFrom
                lTryPreviousFlight = False
                Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " Slot: " & lFrom & " To Early .. Return : " & lReturn)
        Else
            If lMySlotsUsed(lFrom) = -1 Then
                'GOOD the slot is empty
                lReturn = lFrom
                lTryPreviousFlight = False
                Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " Slot empty OK .. Return : " & lReturn)

            Else
                ' the slot is used : try to move the used slot at an  earlier position
                If lTo < 0 Then
                    ' no possible solutions
                    ' we are at the beginning of the slot list without finding a solution
                    lReturn = ReturnNoSolution
                    lTryPreviousFlight = False
                    Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " Slot -- No SOLUTION -- At the end of Slots list Return : " & lReturn)

                Else
                    ' slot used, try to move the flight using the slot to the lTo position
                    ' use recursive call to move the used slot earlier

                    lPos = AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier(lAll_AUFlights, _
                         lMySlotsValueSorted(), lMySlotsUsed(), lSlots_nb, _
                         lTo, lMySlotsUsed(lFrom))

                    If lPos > -1 Then
                        ' found a empty good slot
                        ' make the swap slot
                        lMySlotsUsed(lPos) = lMySlotsUsed(lFrom)
                        lMySlotsUsed(lFrom) = -1

                        lReturn = lFrom 'return the slot put to free by the shift
```

```
                        lTryPreviousFlight = False

                        Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " OK return " & lReturn & _
                                    " Flight move from: " & lFrom & " To: " & lPos)

                    ElseIf lPos = ReturnNoSolution Then
                        ' no solution because flights impossible to move
                        ' cannot create a hole
                        ' Normaly a later solution is possible
                        lReturn = ReturnNoSolution
                        lTryPreviousFlight = False
                        Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " NO SOLUTION ")

                    Else
                        ' a negative value is returned if the position is not movable
                        ' continu to loop with an earlier position

                        'from is initiated and used for next test of ealier slot in this loop
                        lFrom = -lPos + ReturnNotMovableDelta
                        lTo = lFrom - 1

                        If lFrom < 0 Then
                            ' From from next loop must be >= 0 therwhiqse no solution, stop the loop
                            lTryPreviousFlight = False
                            lReturn = ReturnNoSolution

                            Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " NO solution no more Slot to check " & _
                                        lFrom)
                        Else
                            lTryPreviousFlight = True

                            Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " Initial: " & lTargetIx & _
                                        " reloop Next target: " & lFrom)

                        End If

                    End If
                End If
            End If
        End If
    Wend

    Call EX_Log(RecurLevel, "Rec EndLevel : " & lFlightIx_ToMove & " return : " & lReturn)

    AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier = lReturn

    RecurLevel = RecurLevel - 1
End Function


'-------------------------------------------------------------
AU_NPM_ManageMarginPrioFlights_AssignPrioOnlyFlight
' Assign priority only flights in the remaining slot
' list containt prio only + baseline + suspended

Sub AU_NPM_ManageMarginPrioFlights_AssignOtherFlights(lAll_AUFlights As CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer, _
            ByRef lPrioFlightSortedFl() As Integer, lPrioFlight_nb As Integer)

    'don't manage the suspended flight here

    Dim lFl As Integer
    Dim lFlIx As Integer

    Dim lBaselineFlights() As Integer
    Dim lBaselineFlightsNb As Integer

    Dim lPrioOnlyFlights() As Integer
    Dim lPrioOnlyFlightsNb As Integer

    'Dim lSuspendedFlights() As Integer
```



```
    'Dim lSuspendedFlightsNb As Integer


    ReDim lBaselineFlights(lPrioFlight_nb)
    lBaselineFlightsNb = 0

    ReDim lPrioOnlyFlights(lPrioFlight_nb)
    lPrioOnlyFlightsNb = 0

    'ReDim lSuspendedFlights(lPrioFlight_nb)
    'lSuspendedFlightsNb = 0

    For lFl = 0 To lPrioFlight_nb - 1
        lFlIx = lPrioFlightSortedFl(lFl)
        If lAll_AUFlights.GetPrio(lFlIx) = GPrioBaseline Then
            lBaselineFlights(lBaselineFlightsNb) = lFlIx
            lBaselineFlightsNb = lBaselineFlightsNb + 1
        ElseIf lAll_AUFlights.GetPrio(lFlIx) = GPrioSuspended Then
            'lSuspendedFlights(lSuspendedFlightsNb) = lFlIx
            'lSuspendedFlightsNb = lSuspendedFlightsNb + 1
        Else
            lPrioOnlyFlights(lPrioOnlyFlightsNb) = lFlIx
            lPrioOnlyFlightsNb = lPrioOnlyFlightsNb + 1
        End If
    Next lFl


    ' ---------- manage the baseline flights on schedule
    If lBaselineFlightsNb > 0 Then
        Call AU_NPM_ManageMarginPrioFlights_AssignBaselineFlights(lAll_AUFlights, _
                lSlotTime(), lSlotList(), lSlot_nb, _
                lBaselineFlights(), lBaselineFlightsNb)
    End If
    Erase lBaselineFlights


    ' ---------- Manage the prio flights
    If lPrioOnlyFlightsNb > 0 Then
        Call AU_NPM_ManageMarginPrioFlights_AssignPrioFlights(lAll_AUFlights, _
            lSlotTime(), lSlotList(), lSlot_nb, _
            lPrioOnlyFlights(), lPrioOnlyFlightsNb)
    End If
    Erase lPrioOnlyFlights


    '' ---------- Manage the Suspended flights
    'If lSuspendedFlightsNb > 0 Then
    '    Call AU_NPM_ManageMarginPrioFlights_AssignSuspendedFlights(lAll_AUFlights, _
    '        lSlotTime(), lSlotList(), lSlot_nb, _
    '        lSuspendedFlights(), lSuspendedFlightsNb)
    'End If
    'Erase lSuspendedFlights


End Sub


'-------------------------------------------------------------
AU_NPM_ManageMarginPrioFlights_AssignBaselineFlights
' Assign default Baseline flights in the remaining slot
' PB what we do if no possible slot for baseline ??????

Sub AU_NPM_ManageMarginPrioFlights_AssignBaselineFlights(lAll_AUFlights As CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer, _
            ByRef lFlightSorted() As Integer, lFlight_nb As Integer)

    Dim lFl As Integer
    Dim lSlotAssigned As Integer
    Dim lEarliestTime As Date
    Dim lBaselineTime As Date
    Dim lFlAssigned As Integer
    Dim lFlIx As Integer

    ' ---------- manage the baseline flights on schedule
    lFlAssigned = 0
    ' loop on baseline flights
    For lFl = 0 To lFlight_nb - 1
```

```
        lFlIx = lFlightSorted(lFl)
        ' idem part then for Margins
        ' find a slot corresponding to the Margin value to put the flight
        lBaselineTime = lAll_AUFlights.GetBaselineTime(lFlIx)

        lSlotAssigned = AU_NPM_ManageMarginPrioFlights_ManageTimeSolution(lAll_AUFlights, _
                    lSlotTime(), lSlotList(), lSlot_nb, _
                    lBaselineTime, lFlIx)

        If lSlotAssigned < 0 Then
            Call AU_NPM_MsgboxStop("Baseline flight PB of NB of available slot not OK for :
" & _
                    lAll_AUFlights.GetCallsignICAO(lFlIx) & " id= " & lFlIx, _
                    lAll_AUFlights, lSlotTime, lSlotList, lSlot_nb)
        End If
    Next lFl

End Sub




'------------------------------------------------------------
AU_NPM_ManageMarginPrioFlights_AssignPrioOnlyFlight
' Assign priority only flights in the remaining slot

Sub AU_NPM_ManageMarginPrioFlights_AssignPrioFlights(lAll_AUFlights As CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer, _
            ByRef lFlightSorted() As Integer, lFlight_nb As Integer)

    Dim lFl As Integer
    Dim lFlIx As Integer
    Dim lSlotIx As Integer

    Dim lTargetTime As Date
    Dim lSlotAssigned As Integer

    Dim lFlightHaveSolution As Boolean

    ' ---------- Manage the prio flights

    For lFl = 0 To lFlight_nb - 1
        lFlIx = lFlightSorted(lFl)

        ' Manage Prio Flights try to find a free slot compatible with the schedule
        lFlightHaveSolution = False
        For lSlotIx = 0 To lSlot_nb - 1
            If lSlotList(lSlotIx) = -1 Then
                If AU_NPM_ManageMarginPrioFlights_IsFlightScheduleCompatible(lAll_AUFlights,
_
                        lSlotTime(lSlotIx), lFlIx) Then
                    ' the schedule is compatible with the slot time
                    ' lTime = lMySlotsValueSorted(lSlotIx)
                    lSlotList(lSlotIx) = lFlIx

                    'Call lAll_AUFlights.SetFDATime(lFlIx, lSlotTime(lSlotIx))
                    lSlotIx = lSlot_nb ' stop the loop
                    lFlightHaveSolution = True
                End If
            End If
        Next lSlotIx


        ' test if no solution found because slot too early
        If lFlightHaveSolution = False Then
            ' no slot available due to schedule time until the end of the slot list


            lTargetTime = lAll_AUFlights.GetHotspotEndTime

            lSlotAssigned =
AU_NPM_ManageMarginPrioFlights_ManageTimeSolution(lAll_AUFlights, _
                        lSlotTime(), lSlotList(), lSlot_nb, _
                        lTargetTime, lFlIx)

            If lSlotAssigned < 0 Then
```



```
                Call AU_NPM_MsgboxStop("Prio flights PB of NB of available slot not OK for :
" & _
                        lAll_AUFlights.GetCallsignICAO(lFlIx) & " id= " & lFlIx, _
                        lAll_AUFlights, lSlotTime, lSlotList, lSlot_nb)
            End If

        End If
    Next lFl
End Sub

Sub AU_NPM_ManageMarginPrioFlights_AssignSuspendedFlights(lAll_AUFlights As CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer, _
            ByRef lPrioFlightSortedFl() As Integer, lPrioFlight_nb As Integer)

    Dim lFl As Integer
    Dim lFlIx As Integer
    Dim ltime As Date

    ' get the suspended flights
    For lFl = 0 To lPrioFlight_nb - 1
        lFlIx = lPrioFlightSortedFl(lFl)
        If lFlIx > -1 Then ' normally never
            If lAll_AUFlights.GetPrio(lFlIx) = GPrioSuspended Then
                ltime = lAll_AUFlights.GetHotspotEndTime - G_OneSec_AsDate
                Call lAll_AUFlights.SetFDATime(lFlIx, ltime)
            End If
        End If
    Next lFl
End Sub



'------------------------------------------------------------
AU_NPM_ManageMarginPrioFlights_AssignPrioOnlyFlight
' Assign priority only flights in the remaining slot

Sub AU_NPMOLD_ManageMarginPrioFlights_AssignSuspendedFlights(lAll_AUFlights As
CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer, _
            ByRef lFlightSorted() As Integer, lFlight_nb As Integer)

    Dim lFl As Integer
    Dim lFlIx As Integer
    Dim ltime As Date

    ' ---------- manage the baseline flights on schedule
    ' ---------- manage the suspended flights
    For lFl = 0 To lFlight_nb - 1
        lFlIx = lFlightSorted(lFl)
        If lFlIx <> -1 Then
            ' suspended flights at the end of the hotspot
            ' Dont use a slot in the middle
            ' the slot will be use when conpacting at the end

            ltime = lAll_AUFlights.GetHotspotEndTime - G_OneSec_AsDate

            Call lAll_AUFlights.SetFDATime(lFlIx, ltime)
        End If
    Next lFl

End Sub


'------------------------------------------------------
AU_NPM_ManageMarginPrioFlights_UseAvailableSlots
' Assign priority only flights in the remaining slot

Sub AU_NPM_ManageMarginPrioFlights_UseAvailableSlots(lAll_AUFlights As CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer)

    Dim lFl As Integer
    Dim lFlIx As Integer

    Dim lFlChg As Integer
    Dim lFlChgIx As Integer

    Dim lEarliestTime As Date
```

```
983
984     ' compact the list
985     For lFl = 0 To lSlot_nb - 1
986         lFlIx = lSlotList(lFl)
987         If lFlIx = -1 Then
988             ' there is a hole
989             ' find a flight to put here
990             For lFlChg = lFl + 1 To lSlot_nb - 1
991                 lFlChgIx = lSlotList(lFlChg)
992                 If lFlChgIx > -1 Then
993                     ' there is a flight here
994                     If
AU_NPM_ManageMarginPrioFlights_IsFlightScheduleCompatible(lAll_AUFlights, _
995                         lSlotTime(lFl), lFlChgIx) Then
996
997                     'lEarliestTime = lAll_AUFlights.GetBaselineTime(lFlChgIx)
998                     'If lSlotTime(lFl) >= lEarliestTime Then
999                         ' use this flight to fill the hole
1000                        'Call lAll_AUFlights.SetFDATime(lFlChgIx, lSlotTime(lFl))
1001                        lSlotList(lFl) = lFlChgIx
1002                        lSlotList(lFlChg) = -1 ' free the slot
1003                        lFlChg = lSlot_nb ' stop the loop
1004                    End If
1005                End If
1006            Next lFlChg
1007        End If
1008    Next lFl
1009
1010End Sub
1011
1012
```