```
//
(*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++                  Lager
                ++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++
+++  Wolfgang Niedermayr                          V1.5.1
03.02.2018 (V0.0.1) ++++++++++++++++++++++++++++++++++
+++  Abschussprojekt FAAT 2016-2018 +++
Revisions Datum: 07.07.2018 ++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++*)
//

// Hand Auto zuweisungen, Funktionen Sperren und Freigeben.
// Einlagern einer Schublade.
// Speichern von Bauteilen.
// Auslagern.
// Bauteil löschen.
// Bauteil suchen.
// Bereitstellung von Schubladenstatus und Inhalt einer Schublade für HMI.
// Schubladen Status ändern.
// Schubladen Inhalt komplett löschen.
// Ausgabe des gewälten Bauteils an HMI.
// Fehlerauswertung.


//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
//    Init.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
#bAuslagern := #InOut_stVisu.bAuslagern;
#bEinlagern := #InOut_stVisu.bEinlagern;


IF (#IN_bAutomatik) THEN

    IF (#InOut_stVisu.bEinlagern AND NOT #bAuslagernAkt) THEN
        #bEinlagern := true;
    END_IF;

    IF (#InOut_stVisu.bAuslagern AND NOT #bEinlagernStep1Akt) THEN
        #bAuslagern := true;
    END_IF;


ELSIF (#IN_bHandbetrieb) THEN
  // Meldungen generieren kein Automatikbetrieb.
    //  Auslagern.
    IF (#InOut_stVisu.bAuslagern) THEN
        #InOut_Warnungen.%X5 := true;
```

```
    END_IF;
    // Einlagern.
    IF (#InOut_stVisu.bEinlagern) THEN
        #InOut_Warnungen.%X6 := true;
    END_IF;


    #bGefunden := false;
    #bEinlagernStep1Akt := false;
    #bAuslagernAkt := false;
    #bEinlagern := false;
    #bAuslagern := false;

    #InOut_stSteuerung.iLadeNr := #InOut_stVisu.iLadeNr;
    #InOut_stSteuerung.bAuslagern := false;
    #InOut_stSteuerung.bEinlagern := false;
    #InOut_stSteuerung.bLadeHoch := false;

// Keine Betriebsart.
ELSE

    // Meldungen generieren kein Automatikbetrieb.
    IF (#InOut_stVisu.bAuslagern) THEN
        #InOut_Warnungen.%X5 := true;
    END_IF;
    IF (#InOut_stVisu.bEinlagern) THEN
        #InOut_Warnungen.%X6 := true;
    END_IF;

    #InOut_stSteuerung.bAuslagern := false;
    #InOut_stSteuerung.bEinlagern := false;
    #bEinlagernStep1Akt := false;
    #bAuslagernAkt := false;

    #bEinlagern := false;
    #bAuslagern := false;
END_IF;



REGION Einlagern, Speichern


//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
//                              Schublade Einlagern.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
    IF (#bEinlagern AND NOT #arrReFeMerker[1]) THEN

        // Status Ausgelagert.
        IF (#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus = 0) THEN

    // Bauteil nicht hoch.
```

```
//_____________________________________________________________________________
_
            IF (NOT #InOut_stVisu.udtBauteil.bBauteil_Hoch) THEN

            // Schubladenstatus Hoch reset.
                #InOut_stSteuerung.bLadeHoch := false;

            // Lade Übergeben.
                #InOut_stSteuerung.iLadeNr :=
#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iPosition;

            // Einlagern Starten.
                #InOut_stSteuerung.bEinlagern := true;
                #bEinlagernStep1Akt := true;
            ELSE

        // Bauteil ist hoch.

//_____________________________________________________________________________
_
        // Mögliche abzweigungen
            // * Schubladen (HMI 31,32).
            // * Schubladen (HMI 4-30).
            // * Ab Lade 18 Teilung in Vorne und Hinten.


                CASE (#InOut_stVisu.udtBauteil.iLadeNr) OF
                // Schubladen 31,32.
                    31, 32:
                    // Lade Übergeben.
                        #InOut_stSteuerung.iLadeNr :=
#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iPosition;

                    // Schubladenstatus Hoch reset.
                        #InOut_stSteuerung.bLadeHoch := false;

                    // Einlagern Starten.
                        #InOut_stSteuerung.bEinlagern := true;
                        #bEinlagernStep1Akt := true;

                // Restliche Schubladen.
                    ELSE
                    // Teilung Vorne Hinten.

                        // Offset für zu sperrende Schubalde bestimmen.
                        IF (#InOut_stVisu.udtBauteil.iLadeNr > 18) THEN
                            #iStatus_Offfset := 2;
                        ELSE
                            #iStatus_Offfset := 1;
                        END_IF;

                        // Obere Schublade darf nicht aktiv oder voll sein.
                        IF ((#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr +
```

```
#iStatus_Offfset].iStatus <> 1)
                            AND (#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr
+ #iStatus_Offfset].iStatus <> 3)) THEN
        // Starten des Einlagerns.
        // _________________________

                        // Laden Nr. Übergeben.
                            #InOut_stSteuerung.iLadeNr :=
#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iPosition;

                        // Einlagern Starten.
                            #InOut_stSteuerung.bEinlagern := true;
                            #bEinlagernStep1Akt := true;

                        // Schubladenstatus auf Hoch setzen.
                            #InOut_stSteuerung.bLadeHoch := true;
                        ELSE
         // Fehler meldungen.
                        // Bauteil Hoch und Lade Oberhalb ist Eingelegt.
                            #InOut_Warnungen.%X2 := true;
                        END_IF;
                END_CASE;
            END_IF;
        ELSE
        // Fach ist nicht frei oder aktiv.
            #InOut_Warnungen.%X7 := true;
        END_IF;
    END_IF;

// Flankenmerker Aktualisieren.
    #arrReFeMerker[1] := #bEinlagern;


    // Kommunikation mit FB_Auto.
    //_____________________________________________
    IF (#bEinlagernStep1Akt) THEN

        // Falls Bauteil hoch ohne Freigabe des Lagers.
        // oder Maximal Höhe überschritten.
        IF (#InOut_stSteuerung.bFehlerBTHoch) THEN

        //Reset.
            #bEinlagernStep1Akt := false;
            #bEinlagern := false;

            #InOut_stSteuerung.bLadeHoch := false;
            #InOut_stSteuerung.iLadeNr := 255;
            #InOut_stSteuerung.bEinlagern := false;
        END_IF;

        // Einlagern fertig.
        IF (#InOut_stSteuerung.bDone AND NOT #InOut_stSteuerung.bFehlerBTHoch)
THEN
        //Lade Einchecken Prüfen ob Voll und ggf obere Schublade sperren.
```

```
        //
            // * Schublade Hoch.
            // * nicht Hoch oder Nr.31,32.
            IF (#InOut_stSteuerung.bLadeHoch) THEN

            // Prüfen ob Lade Voll.
                #iIndexGefunden := "FC_Seeker"(In_Schublade :=
#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr], In_sGesucht := 'Leer');

                IF (#iIndexGefunden <> 255) THEN
                    #arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus :=
1;
                ELSE
                    #arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus :=
3;
                END_IF;

            // Teilung auf Vorne Hinten.
                IF (#InOut_stVisu.udtBauteil.iLadeNr > 18) THEN
                    #iStatus_Offfset := 2;
                ELSE
                    #iStatus_Offfset := 1;
                END_IF;

            // Schublade oberhalb sperren.
                #arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr +
#iStatus_Offfset].iStatus := 2;

        // Bauteil nicht hoch oder Schublade 31, 32 gewählt.
            ELSE

            // Prüfen ob Lade Voll.
                #iIndexGefunden := "FC_Seeker"(In_Schublade :=
#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr], In_sGesucht := 'Leer');

                IF (#iIndexGefunden <> 255) THEN
                    #arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus :=
1;
                ELSE
                    #arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus :=
3;
                END_IF;
            END_IF;
        // Reset.
        //_______________________________
            //Signal an Visu.
            #InOut_stVisu.bEingelagert := true;

            #bEinlagernStep1Akt := false;

            #InOut_stSteuerung.bLadeHoch := false;
            #InOut_stSteuerung.iLadeNr := 255;
            #InOut_stSteuerung.bEinlagern := false;
            #bEinlagern := false;
```

```
        END_IF;
    END_IF;




//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
//                  Bauteil Anlegen.
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
    IF (#InOut_stVisu.bBauteilAnlegen AND NOT #arrReFeMerker[5]) THEN

    // Bezeichnung nicht leer.
        IF (#InOut_stVisu.udtBauteil.sBezeichnung <> 'Leer') AND
(#InOut_stVisu.udtBauteil.sBezeichnung <> '') THEN

        // Bauteil nicht Hoch.
        //_____________________________________________________________
            IF NOT (#InOut_stVisu.udtBauteil.bBauteil_Hoch) THEN

            // Weiterer Verlauf je nach Schubladen Status.
            // * Frei oder Aktiv.
            // * Gesperrt oder Voll.
                CASE #arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus OF

                    // Schublade nicht Eingelagert oder Eingelegt.
                    0, 1:

                    // Speichern des Bauteils.
                        #iLadeNr :=
#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iNummer;
                        #iMaxSize := #InOut_stVisu.stSchublade.iArrayMaxSize;

                        FOR #ipos := 0 TO #iMaxSize DO
                            IF
(#arrSchubladen[#iLadeNr].arrInhalt[#ipos].sBezeichnung = 'Leer') THEN

                                // Bauteil in Lager Speichern.
                                    #arrSchubladen[#iLadeNr].arrInhalt[#ipos] :=
#InOut_stVisu.udtBauteil;

#arrSchubladen[#iLadeNr].arrInhalt[#ipos].iIndexBauteil := INT_TO_USINT(#ipos);

                                // Rückmeldung.
                                    #InOut_stVisu.bEingelagert := true;

                                // Prüfen ob Schublade voll ist.
                                    IF (#arrSchubladen[#iLadeNr].iStatus = 1) THEN
                                        IF (#ipos = #iMaxSize) THEN
                                            #arrSchubladen[#iLadeNr].iStatus_old :=
#arrSchubladen[#iLadeNr].iStatus;
                                            #arrSchubladen[#iLadeNr].iStatus := 3;
```

```
                                    END_IF;
                                END_IF;
                                // Schleife verlassen.
                                EXIT;
                            END_IF;

                            // Schublade ist voll. Bauteil wurde nicht
eingelagert.
                            IF ((#ipos = #iMaxSize) AND NOT
#InOut_stVisu.bEingelagert) THEN
                                IF (#arrSchubladen[#iLadeNr].iStatus = 1) THEN
                                    #arrSchubladen[#iLadeNr].iStatus_old :=
#arrSchubladen[#iLadeNr].iStatus;
                                    #arrSchubladen[#iLadeNr].iStatus := 3;
                                END_IF;

                            // Warnung ausgeben.
                                #InOut_Warnungen.%X3 := true;
                                #InOut_stVisu.bEingelagert := FALSE;
                            END_IF;
                        END_FOR;

                    // Lade Ist Gesperrt.
                    //
                    2:
                        #InOut_Warnungen.%X4 := true;
                        #InOut_stVisu.bEingelagert := FALSE;


                    // Lade Voll.
                    //
                    3:
                        #InOut_Warnungen.%X3 := true;
                        #InOut_stVisu.bEingelagert := FALSE;

                END_CASE;

        // Bauteil ist Hoch.
        //_____________________________________________
        //
            ELSE
                // * Lade Nr 31,32.
                // * Laden 1,2,3.
                // * Restliche Laden.
                CASE #InOut_stVisu.udtBauteil.iLadeNr OF

                   // Lade 31, 32.
                    31, 32:

                    // Speichern.
                        #iLadeNr := #InOut_stVisu.udtBauteil.iLadeNr;
                        #iMaxSize := #InOut_stVisu.stSchublade.iArrayMaxSize;

                        FOR #ipos := 0 TO #iMaxSize DO
```

```
                            IF
(#arrSchubladen[#iLadeNr].arrInhalt[#ipos].sBezeichnung = 'Leer') THEN

                            // Bauteil in Lager Speichern.
                                #arrSchubladen[#iLadeNr].arrInhalt[#ipos] :=
#InOut_stVisu.udtBauteil;

#arrSchubladen[#iLadeNr].arrInhalt[#ipos].iIndexBauteil := INT_TO_USINT(#ipos);

                                #InOut_stVisu.bEingelagert := true;

                                IF (#arrSchubladen[#iLadeNr].iStatus = 1) THEN
                                    IF (#ipos = #iMaxSize) THEN
                                        #arrSchubladen[#iLadeNr].iStatus_old :=
#arrSchubladen[#iLadeNr].iStatus;
                                        #arrSchubladen[#iLadeNr].iStatus := 3;
                                    END_IF;
                                END_IF;
                                // Schleife verlassen.
                                EXIT;
                            END_IF;
                            // Schublade ist voll. Bauteil wurde nicht
eingelagert.
                            IF ((#ipos = #iMaxSize) AND NOT
#InOut_stVisu.bEingelagert) THEN
                                IF (#arrSchubladen[#iLadeNr].iStatus = 1) THEN
                                    #arrSchubladen[#iLadeNr].iStatus_old :=
#arrSchubladen[#iLadeNr].iStatus;
                                    #arrSchubladen[#iLadeNr].iStatus := 3;
                                END_IF;

                                // Warnung ausgeben.
                                #InOut_Warnungen.%X3 := true;
                                #InOut_stVisu.bEingelagert := FALSE;
                            END_IF;
                        END_FOR;

                    // 1,2,3 Dürfen nicht hoch sein.
                    1,2,3:
                        #InOut_Warnungen.%X8 := true;
                        #InOut_stVisu.bEingelagert := FALSE;

                  // Restliche Schubladen.
                    ELSE
                        // Vorne Hinten Aufteilen.
                        IF (#InOut_stVisu.udtBauteil.iLadeNr > 18) THEN
                            #iStatus_Offfset := 2;
                        ELSE
                            #iStatus_Offfset := 1;
                        END_IF;

                        // Status der Schublade oberhalb frei oder gesperrt.
                        IF (#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr +
#iStatus_Offfset].iStatus = 0)
```

```
                            OR (#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr
+ #iStatus_Offfset].iStatus = 2) THEN

                        //Speichern.
                            #iLadeNr := #InOut_stVisu.udtBauteil.iLadeNr;
                            #iMaxSize :=
#InOut_stVisu.stSchublade.iArrayMaxSize;

                            FOR #ipos := 0 TO #iMaxSize DO
                                IF
(#arrSchubladen[#iLadeNr].arrInhalt[#ipos].sBezeichnung = 'Leer') THEN

                                // Bauteil in Lager Speichern.
                                    #arrSchubladen[#iLadeNr].arrInhalt[#ipos] :=
#InOut_stVisu.udtBauteil;

#arrSchubladen[#iLadeNr].arrInhalt[#ipos].iIndexBauteil := INT_TO_USINT(#ipos);

                                    #arrSchubladen[#iLadeNr +
#iStatus_Offfset].iStatus := 2;

                                    #InOut_stVisu.bEingelagert := true;

                                    IF (#arrSchubladen[#iLadeNr].iStatus = 1)
THEN
                                        IF (#ipos = #iMaxSize) THEN
                                            #arrSchubladen[#iLadeNr].iStatus_old
:= #arrSchubladen[#iLadeNr].iStatus;
                                            #arrSchubladen[#iLadeNr].iStatus :=
3;
                                        END_IF;
                                    END_IF;
                                    // Schleife verlassen.
                                    EXIT;
                                END_IF;

                                // Bauteil wurde nicht eingelagert.
                                IF ((#ipos = #iMaxSize) AND NOT
#InOut_stVisu.bEingelagert) THEN

                                // Schubladen Status ändern.
                                    #arrSchubladen[#iLadeNr].iStatus_old :=
#arrSchubladen[#iLadeNr].iStatus;
                                    #arrSchubladen[#iLadeNr].iStatus := 3;

                                // Warnung ausgeben.
                                    #InOut_Warnungen.%X3 := true;
                                    #InOut_stVisu.bEingelagert := FALSE;
                                END_IF;
                            END_FOR;
                        ELSE

                        //Einzulagerndes Bauteil ist hoch und Schublade oberhalb
ist eingelegt.
```

```
                            #InOut_Warnungen.%X2 := true;
                            #InOut_stVisu.bEingelagert := FALSE;
                        END_IF;
                END_CASE;
            END_IF;
        ELSE

        //Bauteilbezeichnung oder Bauteil nicht angegeben.
            #InOut_Warnungen.%X0 := true;
            #InOut_stVisu.bEingelagert := FALSE;
        END_IF;
    END_IF;


//Flankenmerker.
    #arrReFeMerker[5] := #InOut_stVisu.bBauteilAnlegen;

END_REGION




REGION Auslagern, Ausgeben

//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
//               Schublade Auslagern.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
    IF (#bAuslagern AND NOT #arrReFeMerker[0]) THEN

    // Status Aktiv oder Voll.
        IF (#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus = 1) OR
(#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus = 3) THEN

        //Meldung an Steuerung wo Bauteil Liegt.
            #InOut_stSteuerung.iLadeNr :=
#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iPosition;

        // Auslagern starten.
            #InOut_stSteuerung.bAuslagern := True;

        // Internes Auslagern Bit setzen.
            #bAuslagernAkt := true;
        ELSE

        //Meldung Lade nicht vorhanden oder gesperrt.
            #InOut_Warnungen.%X1 := true;
            #bAuslagernAkt := false;
        END_IF;
    END_IF;
```

```
    //Zurücksetzen der Suchvariablen.
    IF ( #InOut_stSteuerung.bDone AND NOT #InOut_stSteuerung.bBussy AND
#bAuslagernAkt) THEN

    // Aus Lager Auschecken.
        #arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus := 0;
        #arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus_old := 0;

    // Reset.
        #bGefunden := FALSE;
        #InOut_stSteuerung.iLadeNr := 255;
        #InOut_stSteuerung.bAuslagern := false;
        #bAuslagernAkt := false;


    END_IF;


// Flankenmerker zuweisen.
    #arrReFeMerker[0] := #bAuslagern;
    #arrReFeMerker[4] := #InOut_stSteuerung.bDone;

END_REGION




 // Cleanup.
(*
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
//Schubladen Cleanup        Wird bei Bauteil Löschen durchgeführt
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
    IF (#InOut_stVisu.bCleanUp AND NOT #arrReFeMerker[2]) THEN
        FOR #i := 1 TO 32 DO
                #k := 0;
       //Bauteil Array Index init
                FOR #j := 0 TO 23 DO
                    IF NOT (#arrSchubladen[#i].arrInhalt[#j].sBezeichnung =
'Leer') THEN    //Bauteil vorhanden?
                        #arrBT[#k] := #arrSchubladen[#i].arrInhalt[#j];
       //Kopieren nach Temp Arr
                        #k := #k + 1;
       //Index erhöhen
                    END_IF;
                    #arrSchubladen[#i].arrInhalt[#j].iIndexBauteil :=
INT_TO_USINT(#j); //Bauteile neu Indexieren
                END_FOR;

            #arrSchubladen[#i].arrInhalt := #arrBT;
       //Array ohne lücken zurück kopieren ins Lager
            FOR #h := 0 TO 23 DO
                #arrBT[#h] := #udtBauteil;
```

```
       //Temp Array zücksetzen
            END_FOR;
        END_FOR;
    END_IF;


//Flankenmerker
    #arrReFeMerker[2] := #InOut_stVisu.bCleanUp;

*)



//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
//                  Löschen Eines Bauteils und Cleanup.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
IF (#InOut_stVisu.blöschen AND NOT #arrReFeMerker[3]) THEN

    // Löschen des Bauteils.

#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].arrInhalt[#InOut_stVisu.iIndexB
T] := #udtBauteil;
        #ipos := #InOut_stVisu.iLadeNr;

    // Status der Schublade auf alten Status ändern.
        #arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus :=
#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].iStatus_old;

    // Cleanup des Speichers und neu Indexieren.
        // Bauteil Index init.
        #k := 0;
        FOR #j := 0 TO 23 DO
            // Wenn Position Belegt.
            IF NOT (#arrSchubladen[#ipos].arrInhalt[#j].sBezeichnung = 'Leer')
THEN

            // Kopieren in temp Array und Index erhöhen.
                #arrBT[#k] := #arrSchubladen[#ipos].arrInhalt[#j];
                #k := #k + 1;

            END_IF;
        END_FOR;

        // Array ohne lücken zurück kopieren ins Lager.
        #arrSchubladen[#ipos].arrInhalt := #arrBT;


        // Temp Array löschen und Bauteile neu indexieren.
        FOR #h := 0 TO 23 DO

            #arrBT[#h] := #udtBauteil;
            #arrSchubladen[#ipos].arrInhalt[#h].iIndexBauteil :=
```

```
INT_TO_USINT(#h);
        END_FOR;

    // Wechseln der gewählten Seite damit Schubladenseite aktualisiert wird.
        #InOut_stVisu.iLadeNr := #arrSchubladen[#ipos].iPosition;

    // gewähltes Bauteil aktualisiern.
        #InOut_stVisu.bSwichBT := true;
    END_IF;

    #arrReFeMerker[3] := #InOut_stVisu.blöschen;



//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
//                  Löschen des Gesamten Schubladen Inhalts.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
    IF (#InOut_stVisu.bLadeLöschen AND NOT #arrReFeMerker[6]) THEN

    // Inhalt Löschen.
        FOR #o := 0 TO 23 DO
            #arrSchubladen[#InOut_stVisu.iLadeNr].arrInhalt[#o] := #udtBauteil;
        END_FOR;

    // Wechseln der gewählten Seite damit Schubladenseite aktualisiert wird.
        IF (#InOut_stVisu.iLadeNr > 4) THEN
            #InOut_stVisu.iLadeNr := #arrSchubladen[#ipos].iPosition;
        ELSIF (#InOut_stVisu.iLadeNr < 4) THEN
            #InOut_stVisu.iLadeNr := #arrSchubladen[#ipos].iPosition +1;
        END_IF;
    END_IF;


// Flanke zuweisen.
    #arrReFeMerker[6] := #InOut_stVisu.bLadeLöschen;



//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
//                  Schubladen Status ändern.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
    IF (#InOut_stVisu.bStatusWechsel AND NOT #arrReFeMerker[7]) THEN

        #arrSchubladen[#InOut_stVisu.iLadeNr].iStatus :=
#InOut_stVisu.iStatusNeu;
        #arrSchubladen[#InOut_stVisu.iLadeNr].iStatus_old :=
#InOut_stVisu.iStatusNeu;
    END_IF;

// Flanke zuweisen.
```

```
    #arrReFeMerker[7] := #InOut_stVisu.bStatusWechsel;



//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
//                  Gewältes Bauteil an HMI übergeben.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
    IF (#InOut_stVisu.bSwichBT) THEN
        #InOut_stVisu.udtBauteil :=
#arrSchubladen[#InOut_stVisu.udtBauteil.iLadeNr].arrInhalt[#InOut_stVisu.iIndexB
T];
    END_IF;



//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
//                  Gewältes Fach an HMI ausgeben.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
    IF ((#InOut_stVisu.iLadeNr <> #iLadeNr_old)  OR #InOut_stVisu.bSwichLade AND
NOT #arrReFeMerker[9]) THEN

        #InOut_stVisu.stSchublade := #arrSchubladen[#InOut_stVisu.iLadeNr];
    END_IF;

    #arrReFeMerker[9] := #InOut_stVisu.bSwichLade;
    #iLadeNr_old :=#InOut_stVisu.iLadeNr;



//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
//                  Bauteil Suchen.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
    IF(#InOut_stVisu.bBauteilSuchen) THEN
        FOR #l := 1 TO 32 DO

        // Wenn Schublade nicht gesperrt.
            IF (#arrSchubladen[#l].iStatus <> 2)  THEN

            // Suchbaustein Aufrufen.
                IF (#InOut_stVisu.sName <> 'Leer') THEN
                    #iIndexGefunden := "FC_Seeker_TO_Upper"(In_sGesucht :=
#InOut_stVisu.sName, In_Schublade := #arrSchubladen[#l]);
                END_IF;
```

```
            // Rückmeldung an HMI.
                IF (#iIndexGefunden <> 255) THEN

                // Interne Variable Setzen.
                    #bGefunden := true;

                // Meldung an Visu Das Bauteil Gefunden und Ausagerung startet.
                    #InOut_stVisu.bVorhanden := TRUE;
                    #InOut_stVisu.bNichtVorhanden := FALSE;

                // Übergabe des Bauteils.
                    #InOut_stVisu.udtBauteil :=
#arrSchubladen[#l].arrInhalt[#iIndexGefunden];

                // Steuerbit für Bauteilübergabe setzen.
                    #InOut_stVisu.bSwichBT := true;

                // Schleife Verlassen.
                    EXIT;
                END_IF;
            END_IF;

        // Bauteil nicht gefunden.
        // Reset.
            #InOut_stVisu.bVorhanden := false;
            #InOut_stVisu.bNichtVorhanden := true;
        END_FOR;
    END_IF;

// Flanke zuweisen.
    #sName_old := #InOut_stVisu.sName;




// Kopieren.
(*
    IF #strTester.bKopieren THEN
        #arrSchubladen[#strTester.TestBT.iLadeNr].arrInhalt[0] :=
#strTester.TestBT;
        #arrSchubladen[#strTester.TestBT.iLadeNr].iStatus := 1;
    END_IF;
*)




//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
// Init der Schubladen Nummerierung.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
    IF NOT #bInit THEN
        FOR #i := 1 TO 32 DO
```

```
    // Schubladen Positonen Für FB_Auto.
        //Ausgabefach überspringen.
            IF (#i < 4) THEN
                #arrSchubladen[#i].iPosition := INT_TO_USINT(#i);

        //Sprung von 1 auf 2 Seitig.
            ELSIF (#i > 18) THEN
                #arrSchubladen[#i].iPosition := INT_TO_USINT(#i + 2);
            ELSE
                #arrSchubladen[#i].iPosition := INT_TO_USINT(#i + 1);
            END_IF;

     // Nummerierung für Array und HMI zuweisen.
            #arrSchubladen[#i].iNummer := INT_TO_USINT(#i);

    // Indexierung der Bauteile in der Schublade.
            FOR #m := 0 TO 23 DO
                #arrSchubladen[#i].arrInhalt[#m].iIndexBauteil :=
INT_TO_USINT(#m);
                #arrSchubladen[#i].arrInhalt[#m].iLadeNr := INT_TO_USINT(#i);
            END_FOR;
        END_FOR;

    // Init Signal zurücksetzen.
        #bInit := true;
    END_IF;




//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
//              Schubladenstatus an HMI übergeben.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
    FOR #n := 1 TO 32 DO
        #InOut_stVisu.arrSchubladenStatus[#n] := #arrSchubladen[#n].iStatus;
    END_FOR;




//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
//              Meldungen Quittieren.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
//
//  Fehler Beschreibung:
                        (*

                            x0 := Beim Einlagern wurde keine Bauteilbezeichnung
angegeben.
                            X1 := Schublade ist nicht vorhanden oder nicht
```

```
aktiviert.
                            X2 := Bauteil ist hoch und Schublade oberhalb ist
vorhanden.
                            X3 := Schublade ist Voll
                            x4 := Schublade ist Gesperrt
                            x5 := Auslagern und keine Automatik
                            x6 := Einalgern und keine Automatik
                            x7 := Einlagern Schublade bereits aktiv
                            x8 := Schublade 1,2,3 Gesperrt für hoch
                        *)

// Timer zur Quittierung.
    #tQittierung(IN     := #InOut_Warnungen <> 0,
                 PT     := t#5s);

// Qittierung.
    IF (#InOut_Warnungen <> 0) THEN
        #InOut_bSummFehler := true;

        IF ((#IN_bQuitt AND NOT #arrReFeMerker[10]) OR #tQittierung.Q) THEN
            #InOut_Warnungen    := 0;
            #InOut_bSummFehler  := false;
        END_IF;
    END_IF;

// Flanke zuweisen.
    #arrReFeMerker[10] := #IN_bQuitt;
```