```
1
2   % Das ist das Hauptprogramm, wo die Berechnungen stattfinden
3
4   function BgaPtg2Calc(hObject, eventdata, handles, varargin)
5
6   global ticTocFlag;
7
8   % Damit ohne GUI gearbeitet werden kann.
9   global startGuiFlag;
10
11  % Informationen über Komponenten
12  global sheets;
13
14  % Informationen über Prozessketten-Outputs
15  global outTech;
16  global outCost;
17
18  % Beinhaltet die Konstanten der aktuellen Prozesskette
19  global const;
20
21  % Informationen über die Dateien
22  global fileInfos;
23
24  % Interne Parameter
25  global inParam;
26  inParam = struct();
27
28  % Mächtigkeit des Vektors costCompAnnuities (output von Funktion BgaPtg2CalcCompCost)
29  inParam.nrOfCostAnnuities = 6;
30
31  % Aus der Schablone für
32  inParam.nrOfAllComps = 57;
33  inParam.nrOfColOffSet = 2;
34
35  % Cost-Outputs
36  inParam.nrOfCostChainOutputs = 26;
37  inParam.nrOfCostCompOutputs = 9;
38  inParam.nrOfCostOutputs = inParam.nrOfCostChainOutputs + inParam.nrOfAllComps *
    inParam.nrOfCostCompOutputs;
39
40  % Tech-Outputs
41  inParam.nrOfTechChainOutputs = 62;
42  inParam.nrOfTechCompOutputs = 13;
43  inParam.nrOfTechOutputs = inParam.nrOfTechChainOutputs + inParam.nrOfAllComps *
    inParam.nrOfTechCompOutputs;
44
45
46
47  % Ergebniss-Daten für die Outputs in eine Schablone:
48  % Struktur: 'Prozesskette', 'Beschreibung', 'Variablen von outChains & Einheit',
    'Name der Komponente', 'Spalte der Komponente', 'Variablen von outComponents &
    Einheit', ...
49  % (Wiederholung von 4-6 bis aller Komponenten der Prozesskette aufgelsitet sind)
50  % (Wiederholung von zeilen bis alle Prozessketten aufgelsitet sind)
51  xlsCost = {};
52  xlsTech = {};
53
54
55  % Für Error-Vorkommen
56  global errorFlag;
57  errorFlag = false;
58  inParam.errorFlag = false;
59
60  paramFile    = varargin{1};
61  resultFile   = varargin{2};
62  templateFile = which(fileInfos.resultTemplateExcel);
63
64  %% Blatt "Konstante" wird gelesen
65  msg = 'Tabellenblatt "Konstante" wird gelesen.';
66  if startGuiFlag == true
67      BgaPtg2UpdateMessageBox(handles,'apend',msg)
68  else
69      disp(msg)
```

```
70  end
71
72  % Parameter der Komponenten lesen und in Strukturen speichern
73  constSheet = struct();
74  constSheet.value = [];
75  constSheet.txt = {};
76
77  if ticTocFlag == true
78      timeXlsConst = tic;
79  end
80
81  [num, txt, raw]  =  xlsread(paramFile,'Konstante');
82  xRangeConst = size(num,1);
83  if(0 < xRangeConst)
84      % Unnötige Zeilen und Leerzeichen entfernen
85      txt = strtrim(txt(2:1+xRangeConst,1:3));
86      constSheet.value = num(:,1);
87      constSheet.txt   = txt;
88
89      % Fehler in Parameterangabe
90      if((0 < sum(isinf(constSheet.value))) || (0 < sum(isnan(constSheet.value))))
91          errorFlag = true;
92
93      else
94          % Die Konstanten zuweisen
95          const.HHV_CH4               = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'HHV_CH4')));
96          const.LHV_CH4               = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'LHV_CH4')));
97          const.HHV_H2                = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'HHV_H2')));
98          const.LHV_H2                = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'LHV_H2')));
99          const.LHV_kg_H2             = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'LHV_kg_H2')));
100         const.HHV_kg_H2             = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'HHV_kg_H2')));
101         const.HHV_CO2               = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'HHV_CO2')));
102         const.HHV_H2S               = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'HHV_H2S')));
103         const.HHV_H2O               = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'HHV_H2O')));
104         const.HHV_Other             = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'HHV_Other')));
105         const.HHV_TraceGas          = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'HHV_TraceGas')));
106         const.density_CH4           = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'density_CH4')));
107         const.density_CO2           = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'density_CO2')));
108         const.density_H2            = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'density_H2')));
109         const.density_H2S           = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'density_H2S')));
110         const.density_H2O           = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'density_H2O')));
111         const.density_Other         = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'density_Other')));
112         const.density_TraceGas      = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'density_TraceGas')));
113         const.density_Air           = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'density_Air')));
114         const.molarmass_CH4         = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'molarmass_CH4')));
115         const.molarmass_CO2         = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'molarmass_CO2')));
116         const.molarmass_H2          = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'molarmass_H2')));
117         const.molarmass_H2S         = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'molarmass_H2S')));
118         const.molarmass_H2O         = constSheet.value(find(strcmp(constSheet.txt(:,1
            ),'molarmass_H2O')));
```

```
119             const.molarmass_Other       = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'molarmass_Other')));
120             const.molarmass_TraceGas    = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'molarmass_TraceGas')));
121             const.cp_CH4                = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'cp_CH4')));
122             const.cp_CO2                = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'cp_CO2')));
123             const.cp_H2                 = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'cp_H2')));
124             const.cp_H2S                = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'cp_H2S')));
125             const.cp_H2O                = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'cp_H2O')));
126             const.cp_Other              = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'cp_Other')));
127             const.cp_TraceGas           = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'cp_TraceGas')));
128             const.R                     = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'R')));
129             const.convDes               = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'convDes')));
130             const.stoichiometryDes      = constSheet.value(find(strcmp(constSheet.txt(:,1
                ),'stoichiometryDes')));
131         end
132     end
133
134
135     if ticTocFlag == true
136         str = sprintf('Zeit für das Lesen von XLS-Rubric-Konstante: %f Sekunde',toc(
            timeXlsConst));
137         disp(str);
138     end
139
140     % Fehler bei Angabe von Konstanten
141     if(errorFlag == true)
142         msg = 'Fehler im Tabellenblatt "Konstante". Programm wird abgebrochen.';
143         if startGuiFlag == true
144             BgaPtg2UpdateMessageBox(handles,'apend',msg)
145         else
146             disp(msg)
147         end
148         return;
149     end
150
151
152     %% Lese zuerst die Prozesskettenparameter
153     msg = 'Tabellenblatt "Prozesskettenparameter" wird gelesen.';
154     if startGuiFlag == true
155         BgaPtg2UpdateMessageBox(handles,'apend',msg)
156     else
157         disp(msg)
158     end
159
160     if ticTocFlag == true
161         timeChains = tic;
162     end
163
164
165     [numChain, txtChain, rawChain]  =  xlsread(paramFile,'Prozesskettenparameter');
166     [xRangeChainParam,yRangeChainParam] = size(numChain);
167
168     % Unnötige Zeilen und Leerzeichen entfernen
169     txtChainParamInfo = strtrim(txtChain(5:4+xRangeChainParam,1:3));
170     %txtChainName      = strtrim((txtChain(1,4:size(rawChain,2))))';
171     txtChainName      = strtrim((txtChain(1,4:end)))';
172     %valueChainParam   = cell2mat(rawChain(5:4+xRangeChainParam,4:size(rawChain,2)));
173     valueChainParam   = cell2mat(rawChain(5:4+xRangeChainParam,4:size(txtChain,2)));
174
175
176     if ticTocFlag == true
177         str = sprintf('Zeit für das Lesen von XLS-Rubrik-Prozesskette: %f Sekunde',toc(
            timeChains));
```

```
178         disp(str);
179     end
180
181
182     if ticTocFlag == true
183         timeChainsParam = tic;
184     end
185     %% Lese zuerst die Prozesskette
186     msg = 'Tabellenblatt "Prozesskette" wird gelesen.';
187     if startGuiFlag == true
188         BgaPtg2UpdateMessageBox(handles,'apend',msg)
189     else
190         disp(msg)
191     end
192
193     [num, txt, raw]  =  xlsread(paramFile,'Prozesskette');
194
195     xRange = size(num,1);
196
197     % Unnötige Zeilen und Leerzeichen entfernen
198     txt = strtrim(txt(5:4+xRange,1:2));
199
200     % Unnötige Zeilen bei Spalte "Bei Berechnung verwenden?" entfernen
201     pMatrixValue = [];
202     pMatrixTxt   = {};
203     loop = 1;
204     for iCnt = 1:xRange
205         if(num(iCnt,1) == 1)
206             pMatrixValue(loop,:) = num(iCnt,2:end);
207             pMatrixTxt(loop,:) = txt(iCnt,:);
208             loop = loop + 1;
209         end
210     end
211
212     % if ticTocFlag == true
213     %     timeChainsParam = tic;
214     % end;
215     if ticTocFlag == true
216         str = sprintf('Zeit für das Lesen von XLS-Rubrik-Prozesskettenparameter: %f
            Sekunde',toc(timeChainsParam));
217         disp(str);
218     end;
219
220
221
222     % nrOfChains: Die Anzahl der aktiven Prozessketten
223     [nrOfChains,yRange] = size(pMatrixValue);
224
225     % Überprüfen, Die Informationen für die Auswahl der Spalten sind in "sheets"
        gespeichert.
226     % Bei "Biogasinput" darf je Prozesskette nur eine Variante gewählt werden
        (pMatrixValue(:,2:17))
227     % Bei "Methanisieurng" darf je Prozesskette nur eine Variante gewählt werden
        pMatrixValue(:,33:46)
228     % Bei "Gasbereitstellung" darf je Prozesskette nur eine Variante gewählt werden
        pMatrixValue(53,56)
229
230     indexChainFault = [];
231     flagIndexChainFault = false;
232     loop = 1;
233     for iCnt1 = 1:nrOfChains
234         % Restriktionen checken
235         errorFlag = BgaPtg2Restriction(handles, iCnt1, pMatrixTxt, pMatrixValue);
236         if errorFlag == true
237             %msg = ['Fehler in der Analyse von ', char(pMatrixTxt(1,1)), ' -> ',
                char(pMatrixTxt(1,2)), '. Programm wird abgebrochen.'];
238             msg = ['Fehler in der Analyse von ', char(pMatrixTxt(iCnt1,1)), ' -> ', char(
                pMatrixTxt(iCnt1,2))];
239             if startGuiFlag == true
240                 BgaPtg2UpdateMessageBox(handles,'apend',msg)
241             else
242                 disp(msg)
243             end
```

```
244             indexChainFault(loop) = iCnt1;
245             flagIndexChainFault   = true;
246             loop = loop +1;
247         end;
248     end
249
250     % sprintf('%s',cellstr(pMatrixTxt(1,1)))
251     if(flagIndexChainFault == true && size(indexChainFault,2) == nrOfChains)
252         % Es gibt nicht zu brechnen.
253         msg = 'Fehler in der Analyse von allen Prozessketten. Das Programm wird
        abgebrochen.';
254         if startGuiFlag == true
255             BgaPtg2UpdateMessageBox(handles,'apend',msg)
256         else
257             disp(msg)
258         end
259         return;
260     end
261
262     msg = 'Excel-Parameterdatei wird gelesen.';
263     if startGuiFlag == true
264         BgaPtg2UpdateMessageBox(handles,'apend',msg)
265     else
266         disp(msg)
267     end
268
269
270     %% Anfang von Schleifen
271     % *****************************************************************
272     % *****************************************************************
273     % *****************************************************************
274
275     % nrOfChains: Die Anzahl der aktiven Prozessketten
276     %         [nrOfChains,yRange] = size(pMatrixValue);
277     %         inParam.nrOfChains = nrOfChains;
278
279     % Für die Bearbeitung von Zeilen von xlsCost
280     xlsCostRL = 1;
281
282     % Für die Bearbeitung von Zeilen von xlsTech
283     xlsTechRL = 1;
284
285
286     % Sammeln der Annuitäten-Ergebisse aller Prozessketten
287     costChainAnnuityTotalsAll = zeros(inParam.nrOfCostAnnuities,1);
288
289     % Sammeln der Annuitäten auf PK-Ebene
290     costChainAnnuitiesAll = zeros(inParam.nrOfCostAnnuities,1);
291
292     % Für die Speicherung der Verhältnisse der Annuitäten
293     costChainAnnuityTotalRelAll = zeros(inParam.nrOfCostAnnuities+1,1);
294
295
296     % Für die Ergebnisse von Funktion "BgaPtg2TechPreCalc"
297     techSimuPreCalcChainVecAll      = zeros(nrOfChains,8);
298     techSimuPreCalcCompVecAll       = zeros(nrOfChains,10);
299     techPreCalcChainOutputAll       = zeros(nrOfChains,11);
300     techPreCalcChainOutputHelpAll   = zeros(nrOfChains,11);
301
302     % Weitere Ergebnisse
303     techCompSpecialOutput           = zeros(nrOfChains,8);
304     techSimuChainVecAll             = zeros(nrOfChains,7);
305     techSimuOutputVecAll            = zeros(nrOfChains,41);
306
307     costGasLevelisedVecAll          = zeros(nrOfChains,1);
308
309     for iCnt1 = 1:nrOfChains
310
311         if ticTocFlag == true
312             timeChainProzess = tic;
313         end;
314
315
```

```
316         % errorflag zurücksetzen
317         errorFlag = false;
318 
319         % Parameter der Prozessketten lesen und in Strukturen speichern
320         paramChain = struct();
321 
322         % Parameter der Komponenten lesen und in Strukturen speichern
323         paramComp = struct([]);
324 
325         %% Initialisierung von xlsTech
326         xlsTechCL = 1;
327         xlsTech(xlsTechRL,xlsTechCL)    = {pMatrixTxt{iCnt1,1}};
328         xlsTechCL = xlsTechCL + 1;
329         xlsTech(xlsTechRL,xlsTechCL)    = {pMatrixTxt{iCnt1,2}};
330         xlsTechCL = xlsTechCL + 1;
331 
332         % Alle Werte mit null initialisieren
333         for iCntOut = 1:inParam.nrOfTechOutputs
334             xlsTech(xlsTechRL,xlsTechCL) = num2cell(0);
335             xlsTechCL = xlsTechCL + 1;
336         end;
337 
338         % Initialisieren der Tech-Signale auf Chain-Ebene
339         for iCntOut = 1:inParam.nrOfTechChainOutputs
340             str = strcat(outTech{iCntOut,1},'=0;');
341             eval(str);
342         end;
343 
344 
345         %% Initialisierung von xlsCost
346         xlsCostCL = 1;
347         xlsCost(xlsCostRL,xlsCostCL)    = {pMatrixTxt{iCnt1,1}};
348         xlsCostCL = xlsCostCL + 1;
349         xlsCost(xlsCostRL,xlsCostCL)    = {pMatrixTxt{iCnt1,2}};
350         xlsCostCL = xlsCostCL + 1;
351 
352         % Alle Werte mit null initialisieren
353         for iCntOut = 1:inParam.nrOfCostOutputs
354             xlsCost(xlsCostRL,xlsCostCL) = num2cell(0);
355             xlsCostCL = xlsCostCL + 1;
356         end;
357 
358         % Initialisieren der Cost-Signale auf Chain-Ebene
359         for iCntOut = 1:inParam.nrOfCostChainOutputs
360             str = strcat(outCost{iCntOut,1},'=0;');
361             eval(str);
362         end;
363 
364         %% Aktueller Prozesskettenparameter in paramChain speichern
365         % index von Prozesskette in Prozesskettenparameter finden
366         % indexChain = find(strcmp(txtChainName,xlsCost(xlsCostRL,1)));
367         indexChain = find(strcmp(txtChainName,pMatrixTxt{iCnt1,1}));
368 
369         if(1 == size(indexChain,1) && isnumeric(indexChain))
370             paramChain.txtChainParamInfo  = txtChainParamInfo;
371             paramChain.txtChainName       = txtChainName(indexChain);
372             paramChain.valueChainParam    = valueChainParam(:,indexChain);
373 
374             % Fehler in Parameterangabe
375             if((0 < sum(isinf(paramChain.valueChainParam))) || (0 < sum(isnan(paramChain.
                valueChainParam))))
376                 errorFlag = true;
377                 msg = [pMatrixTxt{iCnt1,1} ': Fehler bei der Angabe der Parameter im
                    Tabelletenblatt Prozesskettenparameter.'];
378 
379                 if startGuiFlag == true
380                     BgaPtg2UpdateMessageBox(handles,'apend',msg);
381                 else
382                     disp(msg);
383                 end
384             end
385 
386             % Fehler in Parameterangabe
```

```
387           if(errorFlag == false)
388               for iCnt2 = 1:xRangeChainParam
389                   if(isempty(paramChain.txtChainParamInfo{iCnt2,1}) == 1)
390                       errorFlag = true;
391                       msg = [pMatrixTxt{iCnt1,1} ': Fehler bei der Angabe der Parameter
                          im Tabelletenblatt Prozesskettenparameter.'];
392
393                       if startGuiFlag == true
394                           BgaPtg2UpdateMessageBox(handles,'apend',msg);
395                       else
396                           disp(msg);
397                       end
398                       break;
399                   end
400               end
401           end
402
403       else
404           errorFlag = true;
405           msg = [pMatrixTxt{iCnt1,1} ': Fehler bei der Angabe der Prozesskettennamen im
              Tabelletenblatt Prozesskettenparameter.'];
406
407           if startGuiFlag == true
408               BgaPtg2UpdateMessageBox(handles,'apend',msg);
409           else
410               disp(msg);
411           end
412       end;
413
414        if(errorFlag == false && isempty(find(indexChainFault == iCnt1)))
415
416           indexComponent = [];
417           %% Zweite Schleife über alle Komponenten
418           % ********************************************************************
419           % ********************************************************************
420           % ********************************************************************
421
422           % Index von Komponente feststellen
423           loop = 1;
424           for iCnt2 = 1:yRange
425               if(pMatrixValue(iCnt1,iCnt2) >= 1)
426                   indexComponent(loop) = iCnt2;
427                   loop = loop + 1;
428               end
429           end
430
431           indexComponent = indexComponent';
432
433           % Wieviele Komponenten sind vorhanden?
434           % nrOfComponents: Die Anzahl der aktiven Komponenten
435           nrOfComponents = size(indexComponent,1);
436           inParam.nrOfComponents = nrOfComponents;
437
438           % Sammeln der Anfangsinvestitionen aller Komponenten einer Prozesskette
439           costInvestCompYear1 = zeros(nrOfComponents,1);
440
441           % Sammeln der Ergebisse aller Komponenten einer Prozesskette
442           costCompAnnuities = zeros(nrOfComponents,inParam.nrOfCostAnnuities);
443
444           % Sammeln der Ergebnisse der technischen Simulation
445           if(iCnt1 == 1)
446   %                   techSimuCompVecAll      = zeros(nrOfComponents,9,nrOfChains);
447   %                   techSimuCompValueVecAll = zeros(nrOfComponents,20,nrOfChains);
448   %                   techCompOutputVecAll    = zeros(nrOfComponents,8,nrOfChains);
449   %                   techSimuCompVecAll      = [];
450   %                   techSimuCompValueVecAll = [];
451   %                   techCompOutputVecAll    =
      [];
452                 techSimuCompVecAll      = {};
453                 techSimuCompValueVecAll = {};
454                 techCompOutputVecAll    = {};
455
456           end
```

```
457
458         if ticTocFlag == true
459             timeCompsParam = tic;
460         end;
461
462         % Informationen über die Komponenten lesen
463         for iCnt3 = 1:nrOfComponents
464              %disp (iCnt3);
465              [num, txt, raw]  =  xlsread(paramFile,sheets{indexComponent(iCnt3),2});
466              paramComp(iCnt3).sheet = sheets{indexComponent(iCnt3),2};
467              xRangeComponent = size(num,1);
468              % Unnötige Zeilen und Leerzeichen entfernen
469              txt = txt(10:9+xRangeComponent,1:2);
470              paramComp(iCnt3).valueCompParam = num(:,pMatrixValue(iCnt1,indexComponent
                 (iCnt3)));
471              paramComp(iCnt3).txtCompParamInfo = txt;
472              paramComp(iCnt3).index = indexComponent(iCnt3);
473              paramComp(iCnt3).txtCompName = [sheets{indexComponent(iCnt3),3} ', '
                 sheets{indexComponent(iCnt3),4} ', ' sheets{indexComponent(iCnt3),5} ', '
                  sheets{indexComponent(iCnt3),6}];
474
475              % Fehler in Parameterangabe
476              if((0 < sum(isinf(paramComp(iCnt3).valueCompParam))) || (0 < sum(isnan(
                 paramComp(iCnt3).valueCompParam))))
477                  errorFlag = true;
478                  break;
479              end
480         end
481
482         if ticTocFlag == true
483             str = sprintf('Zeit für das Lesen von XLS-Rubrik-KomponenteXXX: %f
                 Sekunde',toc(timeCompsParam));
484             disp(str);
485         end;
486
487
488         paramComp = paramComp';
489
490         if(errorFlag == true)
491             msg = [pMatrixTxt{iCnt1,1} ': Fehler bei der Angabe der Parameter im
                 Tabelletenblatt ',sheets{indexComponent(iCnt3),2},',',' Set',num2str(
                 pMatrixValue(iCnt1,indexComponent(iCnt3))),'.'];
492
493             if startGuiFlag == true
494                 BgaPtg2UpdateMessageBox(handles,'apend',msg);
495             else
496                 disp(msg);
497             end
498
499         else
500
501             %% Kein Fehler vorhanden
502
503             msg = [pMatrixTxt{iCnt1,1} ': Berechnungen werden durchgeführt.'];
504             if startGuiFlag == true
505                 BgaPtg2UpdateMessageBox(handles,'apend',msg)
506             else
507                 disp(msg)
508             end
509
510             %% Vorberechnungen bei der technischen Simulation wenn notwendig (Anfang
                 von loopPreCalc)
511             for loopPreCalc = 1:nrOfComponents
512                 compColInExcel = paramComp(loopPreCalc).sheet;
513
514                 % Die Biogasinput ist nicht definiert. Daher die
515                 % schleife beenden.
516                 if(compColInExcel == 'U')
517                     break;
518                 end
519
520               if(strcmp(compColInExcel,'E') ||  strcmp(compColInExcel,'F') ||  strcmp(
                  compColInExcel,'G') ||  ...
```

```
521                 strcmp(compColInExcel,'H') ||  strcmp(compColInExcel,'I') ||  strcmp
                    (compColInExcel,'J') ||  ...
522                 strcmp(compColInExcel,'K') ||  strcmp(compColInExcel,'L') ||  strcmp
                    (compColInExcel,'M') ||  ...
523                 strcmp(compColInExcel,'N') ||  strcmp(compColInExcel,'O') ||  strcmp
                    (compColInExcel,'P') ||  ...
524                 strcmp(compColInExcel,'Q') ||  strcmp(compColInExcel,'R') ||  strcmp
                    (compColInExcel,'S') ||  strcmp(compColInExcel,'T'))
525
526
527                     % Outputs definieren
528                     techSimuPreCalcChainVec      = zeros(8,1);
529                     techSimuPreCalcCompVec       = zeros(10,1);
530                     techPreCalcChainOutput       = zeros(11,1);
531                     techPreCalcChainOutputHelp   = zeros(11,1);
532                     techSimuPreCompValueVec      = zeros(10,1);
533
534                     % Funktion zur Simulation der Vorberechnungen aufrufen
535                     [techSimuPreCalcChainVec, techSimuPreCalcCompVec,
                        techPreCalcChainOutput, errorInLoopsFlag] = ...
536                         BgaPtg2TechPreCalc(handles,paramComp,loopPreCalc,paramChain
                        );
537
538                     % Damit von comp-loop rauskommt
539                     if errorInLoopsFlag == true
540                         break;
541                     end
542
543                     % techSimuPreCalcChainVec
544                     techChainVolFlowBG           = techSimuPreCalcChainVec(1);
545                     chainVolFlowCO2              = techSimuPreCalcChainVec(2);
546                     chainVolFlowH2               = techSimuPreCalcChainVec(3);
547                     chainVolFlowH2kg             = techSimuPreCalcChainVec(4);
548                     techChainMethPower           = techSimuPreCalcChainVec(5);
549                     techChainElyPower            = techSimuPreCalcChainVec(6);
550                     techChainpSystem             = techSimuPreCalcChainVec(7);
551                     chainp4Compr                 = techSimuPreCalcChainVec(8);
552
553                     % techSimuPreCalcCompVec
554                     compp                        = techSimuPreCalcCompVec(1);
555                     compT                        = techSimuPreCalcCompVec(2);
556                     compVolFlowBioM              = techSimuPreCalcCompVec(3);
557                     compVolFlowSNG               = techSimuPreCalcCompVec(4);
558                     compVolFlowCO2               = techSimuPreCalcCompVec(5);
559                     compVolFlowH2                = techSimuPreCalcCompVec(6);
560                     compVolFlowH2S               = techSimuPreCalcCompVec(7);
561                     compVolFlowH2O               = techSimuPreCalcCompVec(8);
562                     compVolFlowTraceGas          = techSimuPreCalcCompVec(9);
563                     compVolFlowOther             = techSimuPreCalcCompVec(10);
564
565                     % techSimuPreCalcCompVec
566                     %techChainVolFlowBG           = techPreCalcChainOutput(1);
567                     techChainConcInputCH4        = techPreCalcChainOutput(2);% *
                        100;
568                     techChainConcInputCO2        = techPreCalcChainOutput(3);% *
                        100;
569                     techChainConcInputH2         = techPreCalcChainOutput(4);% *
                        100;
570                     techChainConcInputH2S        = techPreCalcChainOutput(5);% *
                        (10^6);
571                     techChainConcInputH2O        = techPreCalcChainOutput(6);% *
                        100;
572                     techChainConcInputTraceGas   = techPreCalcChainOutput(7);% *
                        (10^6);
573                     techChainConcInputOther      = techPreCalcChainOutput(8);% *
                        100;
574                     %techChainMethPower           = techPreCalcChainOutput(9);
575                     techChainElyPower            = techPreCalcChainOutput(10);
576                     %techChainpSystem             =
                        techPreCalcChainOutput(11);
577
578                     % Ergebnisse sammeln
579                     % wegen Multiplikationen mit 100
```

```
580                         techPreCalcChainOutputHelp      = techPreCalcChainOutput;
581 %                         techPreCalcChainOutputHelp(2)   =
    techPreCalcChainOutputHelp(2) * 100;
582 %                         techPreCalcChainOutputHelp(3)   =
    techPreCalcChainOutputHelp(3) * 100;
583 %                         techPreCalcChainOutputHelp(4)   =
    techPreCalcChainOutputHelp(4) * 100;
584 %                         techPreCalcChainOutputHelp(5)   =
    techPreCalcChainOutputHelp(5) * (10^6);
585 %                         techPreCalcChainOutputHelp(6)   =
    techPreCalcChainOutputHelp(6) * 100;
586 %                         techPreCalcChainOutputHelp(7)   =
    techPreCalcChainOutputHelp(7) * 100;
587 %                         techPreCalcChainOutputHelp(8)   =
    techPreCalcChainOutputHelp(8) * (10^6);
588
589
590                         techSimuPreCalcChainVecAll(iCnt1,:)      =
                            techSimuPreCalcChainVec';
591                         techSimuPreCalcCompVecAll(iCnt1,:)       =
                            techSimuPreCalcCompVec';
592                         techPreCalcChainOutputAll(iCnt1,:)       =
                            techPreCalcChainOutput';
593                         techPreCalcChainOutputHelpAll(iCnt1,:)   =
                            techPreCalcChainOutputHelp';
594
595                         chainVolFlowPG                = techChainVolFlowBG;
596                         techSimuPreCompValueVec       = techSimuPreCalcCompVec;
597                         break;
598                 end;
599             end; % Vorberechnungen bei der technischen Simulation wenn notwendig
                (Anfang von loopPreCalc)
600
601             % Gehe zu nächster Kette
602             if errorInLoopsFlag == true
603                 % Nächste Kette berechnen.
604                 xlsCostRL = xlsCostRL + 1;
605                 xlsTechRL = xlsTechRL + 1;
606                 continue;
607             end
608
609
610             %% Berechnungen der Komponenten für das technische Model (Anfang von
                loopCompTech)
611             techSimuCompVec          = zeros(nrOfComponents,9);
612             techSimuCompValueActVec = zeros(nrOfComponents,20);
613             techCompOutputVec        = zeros(nrOfComponents,8);
614             techCompSpecOutputVec    = zeros(nrOfComponents,9);
615
616             for loopCompTech = 1:nrOfComponents
617
618                 % Initialisieren der Tech-Signale auf Comp-Ebene
619                 for iCntOut = inParam.nrOfTechChainOutputs+1:inParam.
                    nrOfTechCompOutputs
620                     str = strcat(outCost{iCntOut,1},'=0;');
621                     eval(str);
622                 end;
623
624                 test = paramComp(loopCompTech).sheet;
625 %                 if(test == 'BC' || test == 'BF' || test == 'BB')
626 %                     vv = 0;
627 %                 end
628
629                 % Funktion zur Simulation des technischen Teils auf Komponentenebene
630                 % aufrufen
631                 [techSimuCompVec(loopCompTech,:), techSimuCompValueActVec(loopCompTech
                    ,:), techCompOutputVec(loopCompTech,:), techCompSpecOutputVec(
                    loopCompTech,:), errorInLoopsFlag] = ...
632                     BgaPtg2TechSimuComp(handles,paramComp, loopCompTech, paramChain,
                        techSimuPreCompValueVec, techSimuPreCalcChainVec);
633
634                 % Damit von comp-loop rauskommt
635                 if errorInLoopsFlag == true
```

```
636                     break;
637                 end
638
639
640
641                 compPower           = techSimuCompVec(loopCompTech,1);
642                 % techCompFLH        = techSimuCompVec(loopCompTech,2);
643                 techCompElecDemand  = techSimuCompVec(loopCompTech,3);
644                 techCompActCDemand  = techSimuCompVec(loopCompTech,4);
645                 compNutrientDemand  = techSimuCompVec(loopCompTech,5);
646                 techCompHeatDemand  = techSimuCompVec(loopCompTech,6);
647                 compHeatSurplus     = techSimuCompVec(loopCompTech,7);
648                 compWaterDemand     = techSimuCompVec(loopCompTech,8);
649                 techCompOtherDemand = techSimuCompVec(loopCompTech,9);
650
651
652                 compp               = techSimuCompValueActVec(loopCompTech,1);
653                 compT               = techSimuCompValueActVec(loopCompTech,2);
654                 compVolFlowBioM     = techSimuCompValueActVec(loopCompTech,3);
655                 compVolFlowSNG      = techSimuCompValueActVec(loopCompTech,4);
656                 compVolFlowCO2      = techSimuCompValueActVec(loopCompTech,5);
657                 compVolFlowH2       = techSimuCompValueActVec(loopCompTech,6);
658                 compVolFlowH2S      = techSimuCompValueActVec(loopCompTech,7);
659                 compVolFlowH2O      = techSimuCompValueActVec(loopCompTech,8);
660                 compVolFlowTraceGas = techSimuCompValueActVec(loopCompTech,9);
661                 compVolFlowOther    = techSimuCompValueActVec(loopCompTech,10);
662                 compVolFlowPG       = techSimuCompValueActVec(loopCompTech,11);
663                 compConcBioM        = techSimuCompValueActVec(loopCompTech,12);
664                 compConcSNG         = techSimuCompValueActVec(loopCompTech,13);
665                 compConcCO2         = techSimuCompValueActVec(loopCompTech,14);
666                 compConcH2          = techSimuCompValueActVec(loopCompTech,15);
667                 compConcH2S         = techSimuCompValueActVec(loopCompTech,16);
668                 compConcH2O         = techSimuCompValueActVec(loopCompTech,17);
669                 compConcTraceGas    = techSimuCompValueActVec(loopCompTech,18);
670                 compConcOther       = techSimuCompValueActVec(loopCompTech,19);
671                 compDensity_PG      = techSimuCompValueActVec(loopCompTech,20);
672
673
674                 % überschreiben
675                 techSimuPreCompValueVec = [compp; compT; compVolFlowBioM;
                    compVolFlowSNG; compVolFlowCO2; compVolFlowH2; compVolFlowH2S;
                    compVolFlowH2O; compVolFlowTraceGas; compVolFlowOther];
676
677
678                 % Aktueller Wert speichern.
679                 %techSimuPreCompValueVec = techSimuCompValueActVec;
680
681             end; % Berechnungen der Komponenten für das technische Model (Ende von
                loopCompTech)
682
683             % Gehe zu nächster Kette
684             if errorInLoopsFlag == true
685                 % Nächste Kette berechnen.
686                 xlsCostRL = xlsCostRL + 1;
687                 xlsTechRL = xlsTechRL + 1;
688                 continue;
689             end
690
691
692 %                   techSimuCompVecAll(:,:,iCnt1)      = techSimuCompVec;
693 %                   techSimuCompValueVecAll(:,:,iCnt1) = techSimuCompValueActVec;
694 %                   techCompOutputVecAll(:,:,iCnt1)    = techCompOutputVec;
695             techSimuCompVecAll{iCnt1}      = techSimuCompVec;
696             techSimuCompValueVecAll{iCnt1} = techSimuCompValueActVec;
697             techCompOutputVecAll{iCnt1}    = techCompOutputVec;
698
699             %% Weiter technische Berechnung
700             techChainConvCO2Ideal   = 0;
701             techChainConvCO2Real    = 0;
702             techChainComprPower     = 0;
703             techChainNutrientDemand = 0;
704             techChainHeatHT_ideal   = 0;
705             techChainHeatNT_ideal   = 0;
```

```
706             techChainHeatHT_real    = 0;
707             techChainHeatNT_real    = 0;
708             techChainWater4H2       = 0;
709
710             for loop = 1:size(techCompSpecOutputVec,1)

711                 techChainConvCO2Ideal   = techChainConvCO2Ideal +
                    techCompSpecOutputVec(loop,1)*100;
712                 techChainConvCO2Real    = techChainConvCO2Real + techCompSpecOutputVec
                    (loop,2)*100;
713                 if techChainComprPower == 0
714                     %techChainComprPower     = techChainComprPower +
                        techCompSpecOutputVec(loop,3);
715                     techChainComprPower     = techCompSpecOutputVec(loop,3);
716                 end
717                 techChainNutrientDemand = techChainNutrientDemand +
                    techCompSpecOutputVec(loop,4);
718                 techChainHeatHT_ideal   = techChainHeatHT_ideal +
                    techCompSpecOutputVec(loop,5);
719                 techChainHeatNT_ideal   = techChainHeatNT_ideal +
                    techCompSpecOutputVec(loop,6);
720                 techChainHeatHT_real    = techChainHeatHT_real + techCompSpecOutputVec
                    (loop,7);
721                 techChainHeatNT_real    = techChainHeatNT_real + techCompSpecOutputVec
                    (loop,8);
722                 techChainWater4H2       = techChainWater4H2 + techCompSpecOutputVec(
                    loop,9);
723             end;
724
725             techCompSpecialOutput(iCnt1,:) = [techChainConvCO2Ideal,
                techChainConvCO2Real, techChainComprPower, techChainNutrientDemand, ...
726                      techChainHeatHT_ideal, techChainHeatNT_ideal,
                         techChainHeatHT_real, techChainHeatNT_real];
727
728             % Funktion zur Simulation des technischen Teils auf
729             % Kettenebene aufrufen
730             [techSimuChainVec, techSimuOutputVec, errorInLoopsFlag] =
                BgaPtg2TechSimuChain(handles,paramChain,paramComp,techSimuCompVec,
                techSimuCompValueVecAll{iCnt1}, ...
731                                                  techPreCalcChainOutputHelpAll(
                                                     iCnt1,:),techCompSpecialOutput(
                                                     iCnt1,:),techSimuPreCalcChainVec);
732
733             % Gehe zu nächster Kette
734             if errorInLoopsFlag == true
735                 % Nächste Kette berechnen.
736                 xlsCostRL = xlsCostRL + 1;
737                 xlsTechRL = xlsTechRL + 1;
738                 continue;
739             end
740
741             techSimuChainVecAll(iCnt1,:)    = techSimuChainVec';
742             techSimuOutputVecAll(iCnt1,:)   = techSimuOutputVec';
743
744
745
746             %% Kostenmodel
747             % costCompAnnuityAll
748             costCompAnnuityAll    = zeros(nrOfComponents,1);
749
750             % Für die Speicherung der Verhältnisse der Komponenten Annuitäten zur
                Gesamtannuität
751             costCompAnnuityRelAll = zeros(nrOfComponents,1);
752
753             %% Berechnungen der Komponenten für das Kostenmodel (Anfang von
                loopCompCost)
754             for loopCompCost = 1:nrOfComponents
755
756                 % Initialisieren der Cost-Signale auf Comp-Ebene
757                 for iCntOut = inParam.nrOfCostChainOutputs+1:inParam.
                    nrOfCostCompOutputs
758                     str = strcat(outCost{iCntOut,1},'=0;');
759                     eval(str);
```

```
760                 end
761
762                  % Funktion zur Simulation des wirtschaftlichen Teils auf
763                  % Komponentenebene aufrufen
764                  [costCompAnnuities(loopCompCost,:),costInvestCompYear1(loopCompCost),
                     errorInLoopsFlag] = ...
765                      BgaPtg2CalcCompCost(handles, paramChain, paramComp(loopCompCost),
                         techSimuCompVec(loopCompCost,:)',techSimuPreCalcChainVec);
766
767
768                  % Damit von comp-loop rauskommt
769                 if errorInLoopsFlag == true
770                     break;
771                 end
772
773             end    % Berechnungen der Komponenten für das Kostenmodel (Ende von
                    loopCompCost)
774
775             % Gehe zu nächster Kette
776             if errorInLoopsFlag == true
777                 % Nächste Kette berechnen.
778                 xlsCostRL = xlsCostRL + 1;
779                 xlsTechRL = xlsTechRL + 1;
780                 continue;
781             end
782
783
784              % Funktion zur Simulation des wirtschaftlichen Teils auf
785              % Kettenebene aufrufen
786 %
    [costChainAnnuityTotalsAll(:,1),costChainAnnuitiesAll(:,1),costInvestChainYear1,costIn
    vestChainTotalYear1All,costChainAnnuity, ...
787 %
    costChainAnnuityTotalRelAll(:,1),costChainAnnuityHydrogenAll,costHydrogenLCOHAll,costC
    ompAnnuityAll(:,1),costCompAnnuityRelAll(:,1),costChainAnnuityAll,errorInLoopsFlag] =
    ...
788 %
    BgaPtg2CalcChainCost(handles,paramChain,paramComp(1),costCompAnnuities,costInvestCompY
    ear1,techSimuCompVec,techSimuChainVec,techSimuPreCalcChainVec);
789
790
791              [costChainAnnuityTotalsAll(:,1),costChainAnnuitiesAll(:,1),
                 costInvestChainYear1,costInvestChainTotalYear1All,costChainAnnuityTotal,
                 ...
792               costChainAnnuityTotalRelAll(:,1),costChainAnnuityHydrogenAll,
                  costHydrogenLCOHAll,costCompAnnuityAll(:,1),costCompAnnuityRelAll(:,1),
                  costChainAnnuity,errorInLoopsFlag] = ...
793               BgaPtg2CalcChainCost(handles,paramChain,paramComp(1),costCompAnnuities,
                  costInvestCompYear1,techSimuCompVec,techSimuChainVec,
                  techSimuPreCalcChainVec);
794
795
796
797             if(errorInLoopsFlag == true)
798                 msg1 = [pMatrixTxt{iCnt1,1}, ': Fehler bei der Berechnung von
                    chainTechReplacement = ',num2str(inParam.chainTechReplacement),'.'];
799                 msg2 = [pMatrixTxt{iCnt1,1}, ': Es wird von max. 2
                    Ersatzbeschaffungen ausgegangen. Überprüfe Lebensdauer und FLH der
                    Gesamtanlagentechnik'];
800                 if startGuiFlag == true
801                     BgaPtg2UpdateMessageBox(handles,'apend',msg);
802                 else
803                     disp(msg1);
804                     disp(msg2);
805                 end
806             end;
807
808             % Gehe zu nächster Kette
809             if errorInLoopsFlag == true
810                 % Nächste Kette berechnen.
811                 xlsCostRL = xlsCostRL + 1;
812                 xlsTechRL = xlsTechRL + 1;
813                 continue;
```

```
814             end
815
816             % Zuweisungen outputs
817             costChainAnnuityCapitalTotal    = costChainAnnuityTotalsAll(1);
818             costChainAnnuityElecTotal       = costChainAnnuityTotalsAll(2);
819             costChainAnnuityHeatTotal       = costChainAnnuityTotalsAll(3);
820             costChainAnnuityDemandTotal     = costChainAnnuityTotalsAll(4);
821             costChainAnnuityOperationTotal  = costChainAnnuityTotalsAll(5);
822             costChainAnnuityOtherTotal      = costChainAnnuityTotalsAll(6);
823
824             % Zuweisungen outputs
825             costChainAnnuityCapital         = costChainAnnuitiesAll(1);
826             costChainAnnuityElec            = costChainAnnuitiesAll(2);
827             costChainAnnuityHeat            = costChainAnnuitiesAll(3);
828             costChainAnnuityDemand          = costChainAnnuitiesAll(4);
829             costChainAnnuityOperation       = costChainAnnuitiesAll(5);
830             costChainAnnuityOther           = costChainAnnuitiesAll(6);
831
832             % Zuweisungen outputs
833             costChainCapitalTotalRel        = costChainAnnuityTotalRelAll(1);
834             costChainElecTotalRel           = costChainAnnuityTotalRelAll(2);
835             costChainHeatTotalRel           = costChainAnnuityTotalRelAll(3);
836             costChainDemandTotalRel         = costChainAnnuityTotalRelAll(4);
837             costChainOperationTotalRel      = costChainAnnuityTotalRelAll(5);
838             costChainOtherTotalRel          = costChainAnnuityTotalRelAll(6);
839             costChainHydrogenRel            = costChainAnnuityTotalRelAll(7);
840
841             % Funktion BgaPtg2CalcGasCost aufrufen
842             [costGasLevelisedAll, errorInLoopsFlag] = ...
843                 BgaPtg2CalcGasCost(handles, costChainAnnuityTotal, techSimuChainVec,
                    paramChain);
844
845             costGasLevelisedVecAll(iCnt1,1) = costGasLevelisedAll;
846
847             % Gehe zu nächster Kette
848             if errorInLoopsFlag == true
849                 % Nächste Kette berechnen.
850                 xlsCostRL = xlsCostRL + 1;
851                 xlsTechRL = xlsTechRL + 1;
852                 continue;
853             end
854
855
856         end
857
858
859         %% Alle Komponenten sind berechnet
860         %% Eine Schleife der Prozesskette ist berechnet
861         % xlsTech und xlsCost füllen
862         if(errorFlag == false)
863
864             %% Technische Simulation
865             techChainVolFlowBG          = techPreCalcChainOutput(1);
866             techChainConcInputCH4       = techPreCalcChainOutput(2); % * 100;
867             techChainConcInputCO2       = techPreCalcChainOutput(3); % * 100;
868             techChainConcInputH2        = techPreCalcChainOutput(4); % * 100;
869             techChainConcInputH2S       = techPreCalcChainOutput(5); % * 100;
870             techChainConcInputH2O       = techPreCalcChainOutput(6); % * 100;
871             techChainConcInputTraceGas  = techPreCalcChainOutput(7); % * 100;
872             techChainConcInputOther     = techPreCalcChainOutput(8); % * 100;
873             techChainMethPower          = techPreCalcChainOutput(9);
874             techChainElyPower           = techPreCalcChainOutput(10);
875             techChainpSystem            = techPreCalcChainOutput(11);
876             techChainConvCO2Ideal       = techCompSpecialOutput(iCnt1,1);
877             techChainConvCO2Real        = techCompSpecialOutput(iCnt1,2);
878             techChainComprPower         = techCompSpecialOutput(iCnt1,3);
879             techChainNutrientDemand     = techCompSpecialOutput(iCnt1,4);
880             techChainHeatHT_ideal       = techCompSpecialOutput(iCnt1,5);
881             techChainHeatNT_ideal       = techCompSpecialOutput(iCnt1,6);
```

```
882             techChainHeatHT_real          = techCompSpecialOutput(iCnt1,7);
883             techChainHeatNT_real          = techCompSpecialOutput(iCnt1,8);
884
885
886             % Aufrufen der techn. PK Parameter
887             techChainFLH                  = techSimuChainVec(1);
888             techChainEnergySNG            = techSimuChainVec(2);
889             techChainHeatDemand           = techSimuChainVec(3);
890             techChainHeatTotal_real       = techSimuChainVec(4);
891             techChainVolFlowPG            = techSimuChainVec(5);
892             techChainEnergyBioM           = techSimuChainVec(6);
893             techChainElecDemand           = techSimuChainVec(7);
894             %techChainMethPower            = techSimuChainVec(8);
895
896
897             techChainEnergyBioM                 = techSimuOutputVec(1);
898             techChainEnergyCH4                  = techSimuOutputVec(2);
899             techChainVolFlowBioMMix             = techSimuOutputVec(3);
900             techChainVolFlowSNG                 = techSimuOutputVec(4);
901             techChainConcOutputCH4              = techSimuOutputVec(5);
902             techChainConcOutputCO2              = techSimuOutputVec(6);
903             techChainConcOutputH2               = techSimuOutputVec(7);
904             techChainConcOutputH2S              = techSimuOutputVec(8);
905             techChainConcOutputH2O              = techSimuOutputVec(9);
906             techChainConcOutputTraceGas         = techSimuOutputVec(10);
907             techChainConcOutputOther            = techSimuOutputVec(11);
908             techChainHHV_PG                     = techSimuOutputVec(12);
909             techChainRelDensityPG               = techSimuOutputVec(13);
910             techChainWobIdx                     = techSimuOutputVec(14);
911             techChainVolFlowSNGMix              = techSimuOutputVec(15);
912             techChainElecDemandTech             = techSimuOutputVec(16);
913             techChainElecDemandBuild            = techSimuOutputVec(17);
914             techChainHeatDemandTech             = techSimuOutputVec(18);
915             techChainHeatDemandBuild            = techSimuOutputVec(19);
916             techChainElecDemandTotal            = techSimuOutputVec(20);
917             techChainElecDemandCompTotal        = techSimuOutputVec(21);
918             techChainElecDemandMethTotal        = techSimuOutputVec(22);
919             techChainHeatDemandTotal            = techSimuOutputVec(23);
920             techChainHeatDemandCompTotal        = techSimuOutputVec(24);
921             techChainHeatTotal_ideal            = techSimuOutputVec(25);
922             techChainHeatTotal_real             = techSimuOutputVec(26);
923             techChainHeatHT_ideal               = techSimuOutputVec(27);
924             techChainHeatHT_real                = techSimuOutputVec(28);
925             techChainHeatNT_ideal               = techSimuOutputVec(29);
926             techChainHeatNT_real                = techSimuOutputVec(30);
927             techChainEnergyDemandElecHeatTotal  = techSimuOutputVec(31);
928             techChainEffElec                    = techSimuOutputVec(32);
929             techChainEffTotal                   = techSimuOutputVec(33);
930             techChainEffHeatTotal               = techSimuOutputVec(34);
931             techChainEffMeth_elec               = techSimuOutputVec(35);
932             techChainEffMeth_total              = techSimuOutputVec(36);
933             techChainEffMeth_HeatTotal          = techSimuOutputVec(37);
934             techChainPlatzhalter1               = techSimuOutputVec(38);
935             techChainPlatzhalter2               = techSimuOutputVec(39);
936             techChainPlatzhalter3               = techSimuOutputVec(40);
937             techChainPlatzhalter4               = techSimuOutputVec(41);
```

```
938
939
940
941             % Für die Bearbeitung von Spalten von xlsTech
942             xlsTechCL = 1;
943
944             % Prozesskette
945             xlsTech(xlsTechRL,xlsTechCL)    = {pMatrixTxt{iCnt1,1}};
946             xlsTechCL = xlsTechCL + 1;
947
948             % Beschreibung
949             xlsTech(xlsTechRL,xlsTechCL)    = {pMatrixTxt{iCnt1,2}};
950             xlsTechCL = xlsTechCL + 1;
951
952
953             % Chain-Outputs (technische Berechnung)
954             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainVolFlowBG,-2));
955             xlsTechCL = xlsTechCL + 1;
956             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcInputCH4,-2
                ));
957             xlsTechCL = xlsTechCL + 1;
958             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcInputCO2,-2
                ));
959             xlsTechCL = xlsTechCL + 1;
960             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcInputH2,-2
                ));
961             xlsTechCL = xlsTechCL + 1;
962             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcInputH2S,-2
                ));
963             xlsTechCL = xlsTechCL + 1;
964             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcInputH2O,-2
                ));
965             xlsTechCL = xlsTechCL + 1;
966             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(
                techChainConcInputTraceGas,-2));
967             xlsTechCL = xlsTechCL + 1;
968             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcInputOther
                ,-2));
969             xlsTechCL = xlsTechCL + 1;
970             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainMethPower *
                1000,-2));
971             xlsTechCL = xlsTechCL + 1;
972             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainElyPower,-2));
973             xlsTechCL = xlsTechCL + 1;
974             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainFLH,-2));
975             xlsTechCL = xlsTechCL + 1;
976             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainpSystem / (10^5
                ),-2));
977             xlsTechCL = xlsTechCL + 1;
978             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConvCO2Real,-2
                ));
979             xlsTechCL = xlsTechCL + 1;
980             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConvCO2Ideal,-2
                ));
981             xlsTechCL = xlsTechCL + 1;
982             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainComprPower,-2
                ));
983             xlsTechCL = xlsTechCL + 1;
984             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainEnergySNG,-2));
985             xlsTechCL = xlsTechCL + 1;
986             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainEnergyBioM,-2
                ));
987             xlsTechCL = xlsTechCL + 1;
988             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainEnergyCH4,-2));
989             xlsTechCL = xlsTechCL + 1;
990             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainVolFlowSNG,-2
                ));
991             xlsTechCL = xlsTechCL + 1;
992             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainVolFLowBioM,-2
                ));
993             xlsTechCL = xlsTechCL + 1;
994             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainVolFlowPG,-2));
995             xlsTechCL = xlsTechCL + 1;
```

```
996             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainVolFlowSNGMix,-
                2));
997             xlsTechCL = xlsTechCL + 1;
998             xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcOutputCH4,-
                2));
999             xlsTechCL = xlsTechCL + 1;
1000            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcOutputCO2,-
                2));
1001            xlsTechCL = xlsTechCL + 1;
1002            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcOutputH2,-2
                ));
1003            xlsTechCL = xlsTechCL + 1;
1004            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcOutputH2S,-
                2));
1005            xlsTechCL = xlsTechCL + 1;
1006            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcOutputH2O,-
                2));
1007            xlsTechCL = xlsTechCL + 1;
1008            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(
                techChainConcOutputTraceGas,-2));
1009            xlsTechCL = xlsTechCL + 1;
1010            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainConcOutputOther
                ,-2));
1011            xlsTechCL = xlsTechCL + 1;
1012            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainWobIdx,-2));
1013            xlsTechCL = xlsTechCL + 1;
1014            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHHV_PG,-2));
1015            xlsTechCL = xlsTechCL + 1;
1016            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainRelDensityPG,-2
                ));
1017            xlsTechCL = xlsTechCL + 1;
1018            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainEffElec,-2));
1019            xlsTechCL = xlsTechCL + 1;
1020            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainEffTotal,-2));
1021            xlsTechCL = xlsTechCL + 1;
1022            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainEffHeatTotal,-2
                ));
1023            xlsTechCL = xlsTechCL + 1;
1024            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainEffMeth_elec,-2
                ));
1025            xlsTechCL = xlsTechCL + 1;
1026            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainEffMeth_total,-
                2));
1027            xlsTechCL = xlsTechCL + 1;
1028            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(
                techChainEffMeth_HeatTotal,-2));
1029            xlsTechCL = xlsTechCL + 1;
1030            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainElecDemandTotal
                ,-2));
1031            xlsTechCL = xlsTechCL + 1;
1032            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(
                techChainElecDemandCompTotal,-2));
1033            xlsTechCL = xlsTechCL + 1;
1034            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(
                techChainElecDemandMethTotal,-2));
1035            xlsTechCL = xlsTechCL + 1;
1036            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHeatDemandTotal
                ,-2));
1037            xlsTechCL = xlsTechCL + 1;
1038            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(
                techChainHeatDemandCompTotal,-2));
1039            xlsTechCL = xlsTechCL + 1;
1040            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHeatTotal_ideal
                ,-2));
1041            xlsTechCL = xlsTechCL + 1;
1042            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHeatTotal_real
                ,-2));
1043            xlsTechCL = xlsTechCL + 1;
1044            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHeatHT_ideal,-2
                ));
1045            xlsTechCL = xlsTechCL + 1;
1046            xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHeatHT_real,-2
                ));
```

```
1047                xlsTechCL = xlsTechCL + 1;
1048                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHeatNT_ideal,-2
                    ));
1049                xlsTechCL = xlsTechCL + 1;
1050                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHeatNT_real,-2
                    ));
1051                xlsTechCL = xlsTechCL + 1;
1052                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(
                    techChainEnergyDemandElecHeatTotal,-2));
1053                xlsTechCL = xlsTechCL + 1;
1054                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainWater4H2,-2));
1055                xlsTechCL = xlsTechCL + 1;
1056                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainNutrientDemand
                    ,-2));
1057                xlsTechCL = xlsTechCL + 1;
1058                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainPlatzhalter1,-2
                    ));
1059                xlsTechCL = xlsTechCL + 1;
1060                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainPlatzhalter2,-2
                    ));
1061                xlsTechCL = xlsTechCL + 1;
1062                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainPlatzhalter3,-2
                    ));
1063                xlsTechCL = xlsTechCL + 1;
1064                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainPlatzhalter4,-2
                    ));
1065                xlsTechCL = xlsTechCL + 1;
1066                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainElecDemand,-2
                    ));
1067                xlsTechCL = xlsTechCL + 1;
1068                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainElecDemandTech
                    ,-2));
1069                xlsTechCL = xlsTechCL + 1;
1070                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainElecDemandBuild
                    ,-2));
1071                xlsTechCL = xlsTechCL + 1;
1072                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHeatDemand,-2
                    ));
1073                xlsTechCL = xlsTechCL + 1;
1074                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHeatDemandTech
                    ,-2));
1075                xlsTechCL = xlsTechCL + 1;
1076                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techChainHeatDemandBuild
                    ,-2));
1077                xlsTechCL = xlsTechCL + 1;
1078
1079
1080                % Comp-Outputs (technische Berechnung)
1081                for iCnt3 = 1:nrOfComponents
1082    %                       costCompAnnuityCapital          =
      costCompAnnuities(iCnt3,1);
1083    %                       costCompAnnuityElec             =
      costCompAnnuities(iCnt3,2);
1084    %                       costCompAnnuityHeat             =
      costCompAnnuities(iCnt3,3);
1085    %                       costCompAnnuityDemand           =
      costCompAnnuities(iCnt3,4);
1086    %                       costCompAnnuityOperation        =
      costCompAnnuities(iCnt3,5);
1087    %                       costCompAnnuityOther            = costCompAnnuities(iCnt3,6);
1088
1089
1090                    % comp (für alle Komponenten wiederholend, aktuell 57*13 outputs)
1091                    techCompConcCH4                     = techCompOutputVec(iCnt3,1);

1092                    techCompConcCO2                     = techCompOutputVec(iCnt3,2);

1093                    techCompConcH2                      = techCompOutputVec(iCnt3,3);

1094                    techCompConcH2S                     = techCompOutputVec(iCnt3,4);

1095                    techCompConcH2O                     = techCompOutputVec(iCnt3,5);
```

```
1096                techCompConcTraceGas                = techCompOutputVec(iCnt3,6);
1097                techCompConcOther                   = techCompOutputVec(iCnt3,7);
1098                techCompVolFlowPG                   = techCompOutputVec(iCnt3,8);
1099
1100
1101                %compPower                          = techSimuCompVec(iCnt3,1);
1102                techCompFLH                         = techSimuCompVec(iCnt3,2);
1103                techCompElecDemand                  = techSimuCompVec(iCnt3,3);
1104                techCompActCDemand                  = techSimuCompVec(iCnt3,4);
1105                %compNutrientDemand                 =
                    techSimuCompVec(iCnt3,5);
1106                techCompHeatDemand                  = techSimuCompVec(iCnt3,6);
1107                %compHeatSurplus                    =
                    techSimuCompVec(iCnt3,7);
1108                %compWaterDemand                    =
                    techSimuCompVec(iCnt3,8);
1109                techCompOtherDemand                 = techSimuCompVec(iCnt3,9);
1110
1111
1112                xlsTechCL = inParam.nrOfColOffSet + inParam.nrOfTechChainOutputs + (
                    paramComp(iCnt3).index-1)*inParam.nrOfTechCompOutputs + 1;
1113
1114                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompVolFlowPG,-2
                    ));
1115                xlsTechCL = xlsTechCL + 1;
1116                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompConcCH4,-2
                    ));
1117                xlsTechCL = xlsTechCL + 1;
1118                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompConcCO2,-2
                    ));
1119                xlsTechCL = xlsTechCL + 1;
1120                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompConcH2,-2));
1121                xlsTechCL = xlsTechCL + 1;
1122                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompConcH2S,-2
                    ));
1123                xlsTechCL = xlsTechCL + 1;
1124                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompConcH2O,-2
                    ));
1125                xlsTechCL = xlsTechCL + 1;
1126                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompConcTraceGas
                    ,-2));
1127                xlsTechCL = xlsTechCL + 1;
1128                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompConcOther,-2
                    ));
1129                xlsTechCL = xlsTechCL + 1;
1130                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompElecDemand,-
                    2));
1131                xlsTechCL = xlsTechCL + 1;
1132                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompHeatDemand,-
                    2));
1133                xlsTechCL = xlsTechCL + 1;
1134                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompActCDemand,-
                    2));
1135                xlsTechCL = xlsTechCL + 1;
1136                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompOtherDemand
                    ,-2));
1137                xlsTechCL = xlsTechCL + 1;
1138                xlsTech(xlsTechRL,xlsTechCL)    = num2cell(roundn(techCompFLH,-2));
1139                xlsTechCL = xlsTechCL + 1;
1140            end;
1141
1142
1143            %% Kostensimulation
1144
1145            % Für die Bearbeitung von Spalten von xlsCost
1146            xlsCostCL = 1;
1147
1148            % Prozesskette
```

```
1149                xlsCost(xlsCostRL,xlsCostCL)    = {pMatrixTxt{iCnt1,1}};
1150                xlsCostCL = xlsCostCL + 1;
1151
1152                % Beschreibung
1153                xlsCost(xlsCostRL,xlsCostCL)    = {pMatrixTxt{iCnt1,2}};
1154                xlsCostCL = xlsCostCL + 1;
1155
1156
1157                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costGasLevelisedAll,-2
                    ));
1158                xlsCostCL = xlsCostCL + 1;
1159                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costHydrogenLCOHAll,-2
                    ));
1160                xlsCostCL = xlsCostCL + 1;
1161                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainAnnuityTotal,-2
                    ));
1162                xlsCostCL = xlsCostCL + 1;
1163                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                    costChainAnnuityHydrogenAll,-2));
1164                xlsCostCL = xlsCostCL + 1;
1165                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainHydrogenRel,-2
                    ));
1166                xlsCostCL = xlsCostCL + 1;
1167                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                    costInvestChainTotalYear1All,-2));
1168                xlsCostCL = xlsCostCL + 1;
1169                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                    costChainAnnuityCapitalTotal,-2));
1170                xlsCostCL = xlsCostCL + 1;
1171                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                    costChainAnnuityOperationTotal,-2));
1172                xlsCostCL = xlsCostCL + 1;
1173                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                    costChainAnnuityDemandTotal,-2));
1174                xlsCostCL = xlsCostCL + 1;
1175                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                    costChainAnnuityOtherTotal,-2));
1176                xlsCostCL = xlsCostCL + 1;
1177                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                    costChainAnnuityElecTotal,-2));
1178                xlsCostCL = xlsCostCL + 1;
1179                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                    costChainAnnuityHeatTotal,-2));
1180                xlsCostCL = xlsCostCL + 1;
1181                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainCapitalTotalRel
                    ,-2));
1182                xlsCostCL = xlsCostCL + 1;
1183                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                    costChainOperationTotalRel,-2));
1184                xlsCostCL = xlsCostCL + 1;
1185                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainDemandTotalRel
                    ,-2));
1186                xlsCostCL = xlsCostCL + 1;
1187                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainOtherTotalRel,-
                    2));
1188                xlsCostCL = xlsCostCL + 1;
1189                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainElecTotalRel,-2
                    ));
1190                xlsCostCL = xlsCostCL + 1;
1191                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainHeatTotalRel,-2
                    ));
1192                xlsCostCL = xlsCostCL + 1;
1193                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainAnnuityCapital
                    ,-2));
1194                xlsCostCL = xlsCostCL + 1;
1195                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                    costChainAnnuityOperation,-2));
1196                xlsCostCL = xlsCostCL + 1;
1197                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainAnnuityDemand,-
                    2));
1198                xlsCostCL = xlsCostCL + 1;
1199                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainAnnuityOther,-2
                    ));
```

```
1200                xlsCostCL = xlsCostCL + 1;
1201                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainAnnuityElec,-2
                    ));
1202                xlsCostCL = xlsCostCL + 1;
1203                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainAnnuityHeat,-2
                    ));
1204                xlsCostCL = xlsCostCL + 1;
1205                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costChainAnnuity,-2));
1206                xlsCostCL = xlsCostCL + 1;
1207                xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costInvestChainYear1,-2
                    ));
1208                xlsCostCL = xlsCostCL + 1;
1209
1210
1211                for iCnt3 = 1:nrOfComponents
1212
1213                    costCompAnnuityCapital          = costCompAnnuities(iCnt3,1);
1214                    costCompAnnuityElec             = costCompAnnuities(iCnt3,2);
1215                    costCompAnnuityHeat             = costCompAnnuities(iCnt3,3);
1216                    costCompAnnuityDemand           = costCompAnnuities(iCnt3,4);
1217                    costCompAnnuityOperation        = costCompAnnuities(iCnt3,5);
1218                    costCompAnnuityOther            = costCompAnnuities(iCnt3,6);
1219
1220                    xlsCostCL = inParam.nrOfColOffSet + inParam.nrOfCostChainOutputs + (
                        paramComp(iCnt3).index-1)*inParam.nrOfCostCompOutputs + 1;
1221                    xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                        costCompAnnuityCapital,-2));
1222                    xlsCostCL = xlsCostCL + 1;
1223                    xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                        costCompAnnuityOperation,-2));
1224                    xlsCostCL = xlsCostCL + 1;
1225                    xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                        costCompAnnuityDemand,-2));
1226                    xlsCostCL = xlsCostCL + 1;
1227                    xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costCompAnnuityOther
                        ,-2));
1228                    xlsCostCL = xlsCostCL + 1;
1229                    xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costCompAnnuityElec
                        ,-2));
1230                    xlsCostCL = xlsCostCL + 1;
1231                    xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costCompAnnuityHeat
                        ,-2));
1232                    xlsCostCL = xlsCostCL + 1;
1233                    xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costCompAnnuityAll(
                        iCnt3),-2));
1234                    xlsCostCL = xlsCostCL + 1;
1235                    xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(costInvestCompYear1(
                        iCnt3),-2));
1236                    xlsCostCL = xlsCostCL + 1;
1237                    xlsCost(xlsCostRL,xlsCostCL)    = num2cell(roundn(
                        costCompAnnuityRelAll(iCnt3),-2));
1238                    xlsCostCL = xlsCostCL + 1;
1239                end;
1240
1241            end
1242        end;
1243
1244        if ticTocFlag == true
1245            str = sprintf('Zeit für die Ausführung einer Prozesskette: %f Sekunde',toc(
                    timeChainProzess));
1246            disp(str);
1247        end;
1248
1249        % Die nächste Zeile deaktivieren, wenn die Prozesskette und die Komponenten nicht
1250        % nebeneinander geschrieben werden.
1251        xlsCostRL = xlsCostRL + 1;
1252        xlsTechRL = xlsTechRL + 1;
1253
1254    end % end of chain
1255
1256    % Berechnungen sind zu ende. Kein Fehler vorgekommen
1257    % Ergebnisse speichern
1258
```

```
1259  % In Ergebnissdatei schreiben
1260  msg = 'Ergebnisse werden in Excel-Datei geschrieben';
1261  if startGuiFlag == true
1262      BgaPtg2UpdateMessageBox(handles,'apend',msg)
1263  else
1264      disp(msg)
1265  end
1266
1267  if ticTocFlag == true
1268      timeWriteFile = tic;
1269  end;
1270
1271
1272  copyfile(templateFile,resultFile);
1273  area1 = sprintf('A4:ADY%i',nrOfChains+3);
1274  area2 = sprintf('A4:TU%i',nrOfChains+3);
1275
1276
1277  % Formatierung aus matlab
1278  xls = actxserver('Excel.Application');                    % Excel-Server starten
1279
1280  if startGuiFlag == true
1281      wb = xls.Workbooks.Open(resultFile);                  % Datei öffnen
1282  else
1283      wb = xls.Workbooks.Open([pwd() '\' resultFile]);      % Datei öffnen
1284  end
1285
1286
1287  % Results_Technisch
1288  wb.Sheets.Item(2).Activate;                               % 2. Blatt aktivieren
1289  as = wb.ActiveSheet;                                      % Aktives Blatt merken
1290  as.Range(area1).NumberFormat='0,00';
1291
1292  % Results_Wirtschaftlichkeit
1293  wb.Sheets.Item(3).Activate;                               % 3. Blatt aktivieren
1294  as = wb.ActiveSheet;                                      % Aktives Blatt merken
1295  as.Range(area2).NumberFormat='0,00';
1296
1297  wb.Save;                                                  % Blatt sichern
1298  wb.Close;                                                 % Blatt schließen
1299  delete(xls);                                              % Server schließen
1300
1301  xlswrite(resultFile,xlsTech,'Results_Technisch',area1);
1302  xlswrite(resultFile,xlsCost,'Results_Wirtschaftlichkeit',area2);
1303
1304  if ticTocFlag == true
1305      str = sprintf('Zeit für das Schreiben der Ergebnisse: %f Sekunde',toc(
      timeWriteFile));
1306      disp(str);
1307  end;
1308
1309
1310  % Ende des Programms
1311  msg = 'Ende des Programms';
1312  if startGuiFlag == true
1313      BgaPtg2UpdateMessageBox(handles,'apend',msg)
1314  else
1315      disp(msg)
1316  end
1317
1318
```