

```

1
2 #include<iostream>
3 #include <Windows.h>
4 #include<fstream>
5 #include<sstream>
6
7 using namespace std;
8
9 /* Funktionsprototypen*/
10 void auslesen();
11 void vergleich();
12
13 //variablen
14 int resultzahl = 1; //globale zählvariable für die anzahl der erstellten Fehlerdateien
15 //_____
16 int main()
17 {
18     char auswahl = '0';
19     //menu
20     do{
21         cout<<"*****\n";
22         cout<<"*****   PAL Auslese und Vergleichstool V 1.0 *****\n";
23         cout<<"*****   von Mario Roehl *****\n\n\n";
24         cout<<"Auswahl:\n 1 um PAL ueber Serial auszulesen und .pld Datei fuer WNCUPL zu erzeugen.\n 2 um .pld
Dateien zu vergleichen.\n";
25         cout<<" x zum Verlassen\n";
26         cin>>auswahl;
27
28         if(auswahl == '1')auslesen();
29
30         if(auswahl == '2')vergleich();
31
32     }while(auswahl == '1' || auswahl == '2');
33     return 0;
34
35 }// ende Main_____
36
37
38 //__Serial begin_____
39 void auslesen(){
40
41     system("cls");
42     //variablen
43     HANDLE hComm;
44     BOOL m_bPortReady;
45     DCB m_dcb;
46     COMMTIMEOUTS timeouts={0};
47     BOOL m_bStatus;
48     char cTxBuffer[] = "1";
49     char cRxBuffer[40000];
50     unsigned long lBytesWritten, lBytesRead;
51     char datei[50];
52     char ComNr[] = "1";
53     char ComStr[9] = {0};
54     char x[] = "xX";
55     char wdhlg = 'n';
56
57
58     do{ //eingabe komport mit fehler auswertung und exit funktion
59
60         memset(ComStr, '\\0', sizeof(ComStr));
61         cout<<"*****\n";
62         cout<<"*Serial auslesen und in *.pld Datei speichern!*\n";
63         cout<<"*****\n";
64         cout<<"Eingabe Com Port Nr: 1-9, x fuer Ende\n";
65         cin>>ComNr;

```

```

66     if(ComNr[0] == x[0] || ComNr[0] == x[1]){return;}
67
68
69     strcat(ComStr,"//./com");
70     strcat(ComStr,ComNr);
71
72     hComm = CreateFile(ComStr,GENERIC_READ|GENERIC_WRITE,0,0,OPEN_EXISTING,0,0);
73
74     if (hComm == INVALID_HANDLE_VALUE){
75         printf("\nComport Fehler oder Existiert nicht!\n");
76     }
77 }
78 }while(hComm == INVALID_HANDLE_VALUE);
79
80
81 m_bPortReady = GetCommState(hComm, &m_dcb);
82 if (!m_bPortReady){
83     printf("Comport Antwortet nicht!\n");
84     exit(0);
85 }
86
87 m_dcb.BaudRate = 9600;
88 m_dcb.ByteSize = 8;
89 m_dcb.Parity = NOPARITY;
90 m_dcb.StopBits = 1;
91 m_dcb.fBinary = true;
92 m_dcb.fDsrSensitivity = false;
93 m_dcb.fParity = false;
94 m_dcb.fOutX = false;
95 m_dcb.fInX = false;
96 m_dcb.fNull = false;
97 m_dcb.fAbortOnError = true;
98 m_dcb.fOutxCtsFlow = false;
99 m_dcb.fOutxDsrFlow = false;
100 m_dcb.fDtrControl = DTR_CONTROL_DISABLE;
101 m_dcb.fDsrSensitivity = false;
102 m_dcb.fRtsControl = RTS_CONTROL_DISABLE;
103 m_dcb.fOutxCtsFlow = false;
104
105
106 timeouts.ReadIntervalTimeout=50;
107 timeouts.ReadTotalTimeoutConstant=50;
108 timeouts.ReadTotalTimeoutMultiplier=10;
109
110 m_bPortReady = SetCommTimeouts(hComm, &timeouts);
111 if (m_bPortReady != 1){
112     printf("Comport Fehler\n");
113     exit(0);
114 }
115
116
117
118 m_bPortReady = SetCommState(hComm, &m_dcb);
119 if (m_bPortReady != 1){
120     printf("Comport Fehler\n");
121     exit(0);
122 }
123
124 do{
125     //datei abfragen und erstellen
126     cout<<"Dateiname: <dateiname>.pld\n"<<flush;
127     cin>>datei;
128     strcat(datei, ".pld");
129
130     ofstream eingabe(datei,ios_base::out | ios_base::trunc);
131

```

```

132     if(!eingabe.good()){
133         cout<<"Kann Datei nicht Erstellen!";
134         exit(0);
135     }
136
137     cout<<"Datei:  "<<datei<<"    erstellt!\n";
138     m_bStatus = WriteFile(hComm, cTxBuffer, strlen(cTxBuffer),&lBytesWritten,0);
139
140
141     if (m_bStatus)
142     {
143         cout<<"Auslesen gestartet!\n";
144         m_bStatus = ReadFile(hComm,cRxBuffer,sizeof(cRxBuffer)-1,&lBytesRead,0);
145
146         for(unsigned long i =0; i < lBytesRead;i++){
147             if(cRxBuffer[i] == '\n')eingabe<<"\n";
148             eingabe<<cRxBuffer[i];
149         }
150     }
151     cout<<"Datei:  "<<datei<<"    erstellt und geschrieben!\n";
152     eingabe.close();
153
154     cout<<"Weiteren PAL IC auslesen? j/n \n";
155     cin>>wdhlg;
156
157 }while(wdhlg == 'j' || wdhlg == 'J');
158     m_bPortReady = CloseHandle(hComm);
159     return;
160 }
161
162 void vergleich(){
163     system("cls");
164     //variablen
165     HANDLE fHandle;
166     WIN32_FIND_DATA wfd;
167
168     string dateien[50];
169     int dateizaehler = 0;
170     int dateieins = 1;
171     int dateizwei = 2;
172     unsigned long erstegroesse;
173     unsigned long zweitegroesse;
174     char ersteZeilen [250];
175     char zweiteZeilen [250];
176     unsigned long zeilen = 0;
177     bool fehler = false;
178     unsigned long fehlerzahl= 0;
179     bool resultdatei = false;
180     char wdhlg = 'n';
181     char resul[13];
182
183
184     cout<<"*****\n";
185     cout<<"*PLD Dateien vergleichen!*\n";
186     cout<<"*****\n";
187
188     //pld dateien anzeigen und auswählen
189     fHandle=FindFirstFile("*.***",&wfd);
190     FindNextFile(fHandle,&wfd);
191
192     do
193     {
194
195         if (!( (wfd.cFileName[0]=='.') && ( (wfd.cFileName[1]=='.' && wfd.cFileName[2]==0) ||
wfd.cFileName[1]==0 ) ))
196         {

```

```

197 if (wfd.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY){
198
199 }
200 else{
201     string buffer = wfd.cFileName;
202     int laenge = strlen(buffer.c_str());
203     if(buffer[laenge-4]=='.' && buffer[laenge-3]=='p' && buffer[laenge-2]== 'l' &&
buffer[laenge-1]=='d'){
204         dateizaehler++;
205         dateien[dateizaehler] = wfd.cFileName;
206     }
207 }
208 }
209 }
210 while (FindNextFile(fHandle,&wfd) != 0);
211 FindClose(fHandle);
212 //ende datein suchen
213 if(dateizaehler <2){
214     cout<<"Weniger als Zwei .pld Dateien im Ordner gefunden!\n";
215     return ;
216 }
217 do{//wdhlg
218 if(dateizaehler > 2){
219
220 //Dateiliste anzeigen
221
222 for(int i = 1; i <= dateizaehler;i++){
223     cout<<"["<<i<<"] = "<<dateien[i]<<"\n";
224
225 }
226 //Dateien auswählen
227 cout<<"Dateien auswaehlen:\n";
228 cout<<"Erste Datei:";
229 cin>> dateieins;
230 }
231 //erste Datei aussuchen
232 ifstream erste(dateien[dateieins].c_str(),ios::in);
233
234
235 if(!erste.good()){
236     cout<<"Kann Datei:"<<dateien[dateieins]<<" nicht Oeffnen!\n";
237 }
238 else{
239 cout<<dateien[dateieins]<<"\n\n";
240 }
241
242 if(dateizaehler > 2){
243
244 do{
245 cout<<"Zweite Datei:";
246 cin>> dateizwei;
247 if(dateieins == dateizwei){
248     cout<<"zum Vergleichen Bitte Unterschiedliche Dateien auswaehlen!\n";
249 }
250
251 }while(dateieins == dateizwei);
252
253 }
254 //zweite dateie öffnen
255 ifstream zweite(dateien[dateizwei].c_str(),ios::in);
256
257
258 if(!zweite.good()){
259     cout<<"Kann Datei:"<<dateien[dateizwei]<<" nicht Oeffnen!\n";
260 }
261

```

```

262 else{
263 cout<<dateien[dateizwei]<<"\n\n";
264 }
265
266 // ende Dateien aussuchen
267
268 // dateien öffnen und grösse vergleichen
269
270 erste.seekg(0L,ios::end);
271 erstegroesse = erste.tellg();
272
273 zweite.seekg(0L,ios::end);
274 zweitegroesse = zweite.tellg();
275
276 if(erstegroesse == zweitegroesse){
277     cout<<"Dateien gleich Gross!\n";
278 }
279 else{
280     cout<<"Dateien unterschiedlich Gross, Abruch!\n";
281     exit(0);
282 }
283 //datei inhalt vergleichen
284
285 fehlerzahl = 0;
286 zeilen = 1;
287 erste.seekg(0L,ios::beg);
288 zweite.seekg(0L,ios::beg);
289
290 while(!erste.eof() && !zweite.eof()){
291     erste.getline(ersteZeilen,250);
292     zweite.getline(zweiteZeilen,250);
293     zeilen++;
294
295     if(strcmp(ersteZeilen,zweiteZeilen)){
296         fehler = true;
297
298
299         memset(resul, '\0', sizeof(resul));
300         strcat(resul,"result");
301         stringstream ss;
302         ss << resultzahl;
303         strcat(resul,ss.str().c_str());
304         strcat(resul,".txt");
305         if(fehlerzahl<1)ofstream result(resul,ios_base::out | ios_base::trunc);
306         ofstream result(resul,ios_base::out | ios_base::app | ios_base::ate);
307
308         if(!result.good()){ resultdatei = false;}
309         else{
310             resultdatei = true;
311         }
312         result<< "Unterschiedlicher Datei Inhalt in Zeile: "<<zeilen<<"\n";
313         result<<"Datei: "<<dateien[dateieins]<<"\n";
314         result<<ersteZeilen<<"\n";
315         result<<"Datei: "<<dateien[dateizwei]<<"\n";
316         result<<zweiteZeilen<<"\n";
317         result<<"*****\n";
318         fehlerzahl++;
319
320     }//ende Fehlerausgabe bei unterschiedlicher zeile
321 }//ende datei inhaltsvergleich
322
323 if(fehler){
324     cout<<"Dateien haben in: "<<fehlerzahl<<" Zeilen Unterschiedlichen Inhalt!\n";
325
326     if(!resultdatei){
327         cout<<"Konnte keine resultX.txt erstellen!\n";

```

```
328     }
329     else{
330         cout<<resul<<" erstellt!\n";
331     }
332 }
333
334 }
335 else{
336     cout<<"Dateieinhalt ist Identisch!\n";
337 }
338 }
339
340 if(dateizaehler > 2){
341     cout<<"Weitere Dateien vergleichen? j/n\n";
342     cin>>wdhlg;
343 }
344 resultzahl++;
345
346 }while(wdhlg == 'j' || wdhlg == 'J');
347 return ;
348 }
349
```