Anhang B: Dateiformat

Draft

IFW-I Dateiformat

Dateiformatspezifikation für IFW Rastertunnelsysteme
Stand: 16.08.97

Nils Schmeißer
Norbert Matz
Hartwig Hülz

0. Inhalt

1. Motivation

Mit der Implementation neuer Features im Tieftemperatur- und im UHV-System, sowie der Einführung des Auswerteprogramms XRTMA wurde eine Neudefinition und Ablösung des bisherigen IFW-Dateiformates notwendig. Die hier angegebene Empfehlung wurde auf Grundlage des Meßprogrammes von N. Matz/N. Schmeißer sowie des Auswerteprogrammes XRTMA erarbeitet. Sie soll dazu dienen, eine Vereinheitlichung und damit den Austausch von Daten der verschiedenen, im IFW benutzten, STM/AFM Systeme zu ermöglichen.

Draft: Draft bedeutet immer noch nicht Standard. Ein Draft sollte aber möglichst eingehalten werden.

2. Dateiaufbau

Eine Datei besteht aus einem 8 KByte (8192 Byte) großem ASCII-Header, unmittelbar gefolgt von den Daten im Binärformat. Die gültigen Daten des Headers werden durch ein ^Z abgeschlossen. Alle Zeichen zwischen ^Z und dem 8192 Byte Offset sind Füllzeichen.
In einer Datei können mehrere Bilder mit der gleichen Punktgröße vorhanden sein. Für jedes Bild gibt es im Header einen Eintrag. Zusätzlich können Spektroskopiekurven abgelegt werden.

2.1. Header

Konventionen:

<>	Wert des eingeschlossenen Ausdruckes
[]	optionaler Ausdruck
""	Terminale, die von " eingeschlossenen Zeichen müssen so übetragen werden
S_1 | S_2	Alternative, entweder S_1 oder S_2
{}	Klammerung von Ausdrücken

Der Header gliedert sich in mehrere Sektionen, die durch "*****" <Schlüsselwort> eingeleitet werden. Jede Sektion enthält Einträge, die durch "\" eingeleitet werden, gefolgt von einem Schlüsselwort, einem Doppelpunkt mit Leerzeichen ("**:** ") und den zugehörigen Werten (als Text). Jeder Eintrag wird durch CR und LF abgeschlossen.

Sektionen:	*File list	Dateiinformationen
	*Microscope list	Informationen über verwendetes Mikroskop und Messung
	*Controller list	Informationen über SPM-Regelung (optional)
	*Environment	Umgebungscharakteristika
	*Image info	für jedes Teilbild vorhanden, enthält Informationen über das Teilbild

Die Reihenfolge der Sektionen ist strikt einzuhalten.
Der Einheitenvorsatz **µ** ist durch ein **u** zu ersetzen.

2.1.1. *File list

Feld	Syntax der Parameter	Bedeutung
\Date	<hh> ":" <mm> ":" <ss> <dd> "." <mm> "." <yyyy>	Uhrzeit und Datum des Starts der Messung, das Jahr wird als 4-stellige Zahl angegeben
\Data length	"8192"	Offset der Daten, 8192 ist vorgeschrieben
\Text		optionaler Kommentar
\History		optionaler Kommentar

2.1.2. *Microscope list

Feld	Syntax der Parameter	Bedeutung
\Scan size	<size> { "um" | "nm" | "Å" }	Größe in X und Y-Richtung in Einheit
\X offset	<offset> nm	Auslenkung in X-Richtung (nm)
\Y offset	<offset> nm	Auslenkung in Y-Richtung (nm)
\Line direction	{"Trace" | "Retrace" }	Trace: Daten wurden beim Hinlauf aufgenommen, Retrace: beim Rücklauf
\Rotate Ang.	<angle>	Richtung der schnellen Scan-richtung relativ zur Probe, in Altgrad, math. neg. Sinn
\Samps/line	<nx> <ny>	nx: Anzahl Punkte pro Zeile ny: Anzahl Zeilen
\Scan rate	<lps> "1/s"	*lps*: Lines per second, Anzahl Zeilen pro Sekunde
\Sample period	<tbs> " *0.1 us"	*tbs*: Time between sample, Zeit zwischen zwei Meßwert-abgriffen in *1/10* µ *s*
\Int. gain	<igain>	Internal gain
\Prop. gain	<pgain>	Proportional gain
\Filter	<f> "Hz"	0: keine Filter <f>: Tiefpaßfilter mit Grenzfrequenz *f* in *Hz*
\Scope dualtrace	{ "Single" | "Dual" }	Single: nur Hinlauf Dual: auch Rücklauf wird gemessen
\Z sens	<sens>	Verstärkungsfaktor am Hochspannungsverstärker
\ZPiezo type	{ "Atype" | "Ctype" }	Z Piezo Typ
\Z scale	<scale>	Eichfaktor für Z-Piezo
\XYPiezo type	{ "Atype" | "Ctype" }	X und Y Piezo Typ
\X scale	<xscale>	Eichfaktor X
\Y scale	<yscale>	Eichfaktor Y
\Start context	{ "STM" | "AM"	Gerätespezifika: STM - Tunnelmikroskop AFM - Kraftmikroskop
\Id	["LT"] ["UHV"] ["LC"] ["ITC4"]	Mikroskopkontext: LT - Tieftemperaturerw. UHV - UH-Vakuumerw. LC - ITC4- ITC4 Temperatursteuerung
\Mode	{ "CCM" | "CHM" | "CON"}	Meßmodus: CCM - Constant Current M. CHM - Constant Height M. CON - Contact M.

nur STM		
\Bias	<bias> { "uV" | "mV" | "V" }	Tunnelspannung
\Biasfreq.	<f> "Hz"	Frequenz der Tunnelspannung
\Current	<current> { "nA" | "µA" | "mA" }	Tunnelstrom
nur SFM		
\SpingK	<k> N/m	Federkonstante

2.1.3. *Controller list

Feld	Syntax der Parameter	Bedeutung
\Switches Top	<topsw>	Schalterstellung Topographie an der Lehmann-Regelung
\Switches Pot	<potsw>	Schalterstellung Potentiometrie an der Lehmann-Regelung
\DAC U(Piezo X)	<dacout>	DAC Ausgang X Piezo
\DAC U(Piezo Y)	<dacout>	DAC Ausgang X Piezo
\DAC U(Piezo Z)	<dacout>	DAC Ausgang X Piezo
\DAC U(Tunnel)	<dacout>	DAC Ausgang
\ADC U(Pizeo X)	<adcin>	ADC Eingang X Piezo
\ADC U(Pizeo Y)	<adcin>	ADC Eingang Y Piezo
\ADC U(Pizeo Z)	<adcin>	ADC Eingang Z Piezo
\ADC U(Tunnel)	<adcin>	ADC Eingang Tunnelspannung
\ADC I(Tunnel)	<adcin>	ADC Eingang Tunnelstrom
\ADC Topography	<adcin>	ADC Eingang Topographie
\ADC Potentiometry	<adcin>	ADC Eingang Potentiometrie
\ADC Top-Gleichrichter	<adcin>	ADC Eingang Top. Gleichrichter
\ADC samples	<adcin>	ADC Eingang samples
\Lock-in timeconst	<lock1> <lock2>	Lock-In Zeitkonstanten
\Lock-in sens	<sens1> <sens2>	Empfindlichkeiten der Lock-In Verstärker
\Lock-in phase	<ph1> <ph2>	Phase im Lock-In
\Ref voltage	<volt> { "mV" | "V" }	Referenzsp. im Lock-In
\Ref current	<current> { "nA" | "µA" | "mA" }	Referenzstrom im Lock-In
\Ref frequency	<frq> "Hz"	Referenzfreq. im Lock-In
\Pause control	<dly>	Anpassungszeit für Regelung in μs
\Pause piezo	<dly>	Wartezeit vor jeder Messung in *ms*.

2.1.4. *Environment

\Temperature	<temp> "K"	Temperatur in *Kelvin*
\Pressure	<press> "mbar"	Druck in *mbar*

2.1.5. *Image info

Dieser Abschnitt ist für jedes Teilbild vorhanden.

Feld	Syntax der Parameter	Bedeutung
\Image	<n>	Bildnummer
\Doffset	<ofs>	Daten-Offset ab \Data offset -> Daten auf *doffset+ofs*
\Type	{ "TOP" | "POT" | "SPE" | "HAR" | "WFC"}	Bildtyp: TOP - Topographie POT - Potentialverteilung SPE - Spektroskopie HAR - Harmonische WFC - Workfunction
\Z scaling	<zscale> <unit>	Z-Skalierungsfaktor *zscale*
\Z offset	<zoffset> <unit>	$Wert=zscale*s_{i,i}+zoffset$
\Text	<string>	Kommentarzeile
nur POT		
\POT voltage	<upom> "mV"	Potentiometriespannung in *mV*
\POT freq.	<freq> "Hz"	Frequenz in *Hz*
\POT int. gain	<n>	internal Potentiometry gain
nur HAR		
\Degree	<n>	Grad der Harmonischen
\Phase	<phase>	Phasenlage in Altgrad
nur SPE		
\SType	{ "U0" | "DER" | "I_U" | "ISO" }	Typ der Daten U0 - für festes U_0 DER - Ableitung an der Stelle U_0 I_U - I-U Kennlinie ISO - Isofläche *U* für konstantes *I*
\USmin	<usmin> "mV"	minimale Spannung
\USmax	<usmax> "mV"	maximale Spannung
\U0	<u> "mV"	Stelle U_0
\I0	<i> <unit>	nur für Isofläche
\Numsamples	<n>	Anzahl Meßwerte bei I_U
\Point	<x> <y>	Koordinaten (integer, (0,0) = linke obere Ecke) "der I-U Kennlinie"
\Repeat	<n>	Anzahl Meßwerte pro Punkt (Mittelwertberechnung)

2.2. Datenbereich

Jedes Datum ist ein Integer-Wert (2 Byte) nach Spezifikation auf dem MS/DOS-PC (niederwertiges Byte zuerst, 2-er Komplement). Die Daten werden zeilenweise, von links nach rechts abgelegt. Eine Datei kann mehrere Bilder enthalten, die nacheinander, ohne Lücke, gespeichert werden.
Rekonstruktion der Daten aus den Informationen der *Image info Sektion:

$$w_{i,j} = zscale \cdot s_{i,j} + zoffset$$

mit den Koordinaten

$$x = xoffset + j \cdot \frac{size}{n_x - 1} \text{ bzw. } y = yoffset + i \cdot \frac{size}{n_y - 1}.$$

Rekonstruktion von I-U Kennlinien in der Spektroskopie:

$$I_k = s_k \cdot \frac{USmax - USmin}{Numsamples - 1} + USmin.$$

3. Vorgeschlagene Funktionen

```
{
          File: IFWFILE.PAS
        Author: Nils Schmeisser
   Description: Storing data to IFW-I file
}

unit IFWFile;

interface

uses arrays;

const TRACE  =TRUE;
      RETRACE=FALSE;

      CCM    =1;
      CHM    =2;

      TOP    =1;
      POT    =2;
      SPE    =3;
      HAR    =4;

      U0     =1; { data at U=U0 }
      DER    =2; { derivation at U=U0 }
      I_U    =3; { I-U graph during hold and sample }
      ISO    =4; { isograph for I=I0 }

      MAXDATA=30;

      OK     = 0; { everything went O.K. }
      FNOPEN =-1; { File not opened }
      FOPEN  =-2; { File already opened }
      FM2    =-3; { can't open more than one }
      DNALLOC=-4; { data not exist (probably deallocated) }
      NHEAD  =-5; { no header data }
      NDATA  =-6; { no data to store }
      MSUBD  =-7; { not more then MAXDATA subdata allowed }

type timerec=record
               day,month,year:word;
               hour,minute,second,second100:word;
             end;

     timeinfo=record
                start,stop:timerec;
              end;

     ifwfilestruct=record
                     opened:boolean;
                     fname:string;
                     hlcount:integer;
                     header:array[1..128] of string;
                     dcount:integer;
                     data:array[1..MAXDATA] of PArray;
                   end;

procedure NewIFWFile(var fvar:ifwfilestruct);

function OpenIFWFile(fname_:string; var fvar:ifwfilestruct):integer;
```

```
function SetIFWHeader(time:timeinfo;
                      textfield,historyfield:string;
                      { Scan an Microscope infos }
                      nx,ny:word;
                      offsetx,offsety:extended;  { in nm }
                      ssize:extended;            { in following unit }
                      range:integer;             { 10 ^ -10 | -9 | -6 | -3 m}
                      ldir:boolean;              { TRACE/RETRACE }
                      mode:integer;              { CCM or CHM }
                      igain,pgain:integer;       { int. and prop. gain }
                      filter:integer;            { lowpassfilter freq.0=None}
                      dualtrace:boolean;         { Trace and Retrace }
                      u_tunnel,ut_freq:extended; { Bias and Biasfreq. }
                      i_tunnel:extended;         { Tunnelcurrent }
                      zptype,xyptype:char;       { Pizeotypes }
                      EichX,EichY,EichZ:extended;{ Temp. calibration factors}
                      VerstZ:extended;           { amplification }
                      ctx,id:string;             { Mic. context and ID }
                      { Controller }
                      cid:string;                { Controller ID }
                      { Environment }
                      temperature:extended;
                      pressure:extended;
                      var fvar:ifwfilestruct):integer;

function AddIFWData(p:PArray;
                      itype:integer; zscale,zoffset:extended; un:string;
                      textfield:string;
                      { for Potentiometry }
                      upot,upotfreq:extended; upotgain:integer;
                      { Harmonics }
                      degree:integer; phase:extended;
                      { Spectroscopy }
                      stype:byte;       { U0, DER, I_U, ISO }
                      usmin,usmax,us0,is0:extended;
                      nsamp,ix,iy,rep:integer;
                      var fvar:ifwfilestruct):integer;

function CloseIFWFile(var fvar:ifwfilestruct):integer;

implementation

procedure NewIFWFile(var fvar:ifwfilestruct);
begin
  fvar.opened:=false;
end;

function OpenIFWFile(fname_:string; var fvar:ifwfilestruct):integer;
begin
  if (fvar.opened) then begin
    if (fvar.fname=fname_) then
      OpenIFWFile:=FM2
    else
      OpenIFWFile:=FOPEN;
  end else begin
    with fvar do begin
      opened:=TRUE;
      fname:=fname_;
      hlcount:=0;
      dcount:=0;
    end;
    OpenIFWFile:=OK;
  end;
end;

procedure printh(var fvar:ifwfilestruct; s:string);
begin
  inc(fvar.hlcount);
  fvar.header[fvar.hlcount]:=s;
end;
```

```
function AddIFWData(p:PArray;
                    itype:integer; zscale,zoffset:extended; un:string;
                    textfield:string;
                    { for Potentiometry }
                    upot,upotfreq:extended; upotgain:integer;
                    { Harmonics }
                    degree:integer; phase:extended;
                    { Spectroscopy }
                    stype:byte;       { U0, DER, I_U, ISO }
                    usmin,usmax,us0,is0:extended;
                    nsamp,ix,iy,rep:integer;
                    var fvar:ifwfilestruct):integer;
var s:string;
    x,y:string;
    z,sp:word;
    i:integer;
    doffset:longint;
begin
  if not(fvar.opened) then
    AddIFWData:=FNOPEN
  else if (fvar.dcount=MAXDATA) then
    AddIFWData:=MSUBD
  else begin
    printh(fvar,'\*Image info');
    doffset:=0;
    if (fvar.dcount>0) then begin
      for i:=1 to fvar.dcount do begin
        (fvar.data[i])^.GetDim(z,sp);
        doffset:=doffset+z*sp*(fvar.data[i])^.GetSize;
      end;
    end;
    inc(fvar.dcount);
    fvar.data[fvar.dcount]:=p;
    str(fvar.dcount,s);
    printh(fvar,'\Image: '+s);
    str(doffset,s);
    printh(fvar,'\Doffset: '+s);
    case itype of
      TOP: s:='TOP';
      POT: s:='POT';
      SPE: s:='SPE';
      HAR: s:='HAR';
    end;
    printh(fvar,'\Type: '+s);
    str(zscale:0:3,s);
    printh(fvar,'\Z scaling: '+s+' '+un);
    str(zoffset:0:3,s);
    printh(fvar,'\Z offset: '+s+' '+un);
    printh(fvar,'\Text: '+textfield);
    if (itype=POT) then begin
      str(upot:0:3,s);
      printh(fvar,'\POT voltage: '+s+' mV');
      str(upotfreq:0:3,s);
      printh(fvar,'\POT freq.: '+s+' Hz');
      str(upotgain,s);
      printh(fvar,'\POT int. gain: '+s);
    end else if (itype=HAR) then begin
      str(degree,s);
      printh(fvar,'\Degree: '+s);
      str(phase,s);
      printh(fvar,'\Phase: '+s);
    end else if (itype=SPE) then begin
      case stype of
         U0: s:='U0';
        DER: s:='DER';
        I_U: s:='I_U';
        ISO: s:='ISO';
      end;
      printh(fvar,'\SType: '+s);
      str(usmin:0:3,s);
```

```
      printh(fvar,'\USmin: '+s+' mV');
      str(usmax:0:3,s);
      printh(fvar,'\USmax: '+s+' mV');
      str(us0:0:3,s);
      printh(fvar,'\U0: '+s+' mV');
      str(is0:0:3,s);
      printh(fvar,'\I0: '+s+' mV');
      str(nsamp,s);
      printh(fvar,'\Numsamples: '+s);
      str(ix,x);
      str(iy,y);
      printh(fvar,'\Point: '+x+' '+y);
      str(rep,s);
      printh(fvar,'\Repeat: '+s);
    end;
    AddIFWData:=OK;
  end;
end;

function SetIFWHeader( time:timeinfo;
                    textfield,historyfield:string;
                    { Scan an Microscope infos }
                    nx,ny:word;
                    offsetx,offsety:extended;  { in nm }
                    ssize:extended;            { in following unit }
                    range:integer;             { 10 ^ -10 | -9 | -6 | -3 m}
                    ldir:boolean;              { TRACE/RETRACE }
                    mode:integer;              { CCM or CHM }
                    igain,pgain:integer;       { int. and prop. gain }
                    filter:integer;            { lowpassfilter freq.0=None}
                    dualtrace:boolean;         { Trace and Retrace }
                    u_tunnel,ut_freq:extended; { Bias and Biasfreq.forSTM,
                                                 sprinconst.for AFM}
                    i_tunnel:extended;         { Tunnelcurrent }
                    zptype,xyptype:char;       { Pizeotypes }
                    EichX,EichY,EichZ:extended;{ Temp. calibration factors}
                    VerstZ:extended;           { amplification }
                    ctx,id:string;             { Mic. context and ID }
                    { Controller }
                    cid:string;                { Controller ID }
                    { Environment }
                    temperature:extended;
                    pressure:extended;
                    var fvar:ifwfilestruct):integer;
var  s,a1,a2:string;
     i:integer;
     t1,t2,dt:longint;
     edt:extended;
begin
  if not(fvar.opened) then
    SetIFWHeader:=FNOPEN
  else begin
    with time.start do
      t1:=(((longint(day)*24+longint(hour))*60+longint(minute))*60
            +longint(second))*100+longint(second100);
    with time.stop do
      t2:=(((longint(day)*24+longint(hour))*60+longint(minute))*60
            +longint(second))*100+longint(second100);
    dt:=t2-t1;

    {--- File list ----------------------------------------}
    printh(fvar,'\*File list');
    with time.start do begin
      str(hour:2,a2);
      str(minute:2,a1);
      a2:=a2+':'+a1;
      str(second:2,a1);
      a2:=a2+':'+a1+' ';
      str(day:2,a1);
      a2:=a2+a1+'.';
```

```
  str(month:2,a1);
  a2:=a2+a1+'.';
  str(year:4,a1);
  a2:=a2+a1;
end;
printh(fvar,'\Date: '+a2);
printh(fvar,'\Data length: 8192');
printh(fvar,'\Text: '+textfield);
printh(fvar,'\History: '+historyfield);
{--- Microscope list -------------------------------------}
printh(fvar,'\*Microscope list');
str(ssize:0:1,a1);
a1:=a1+' ';
case range of
 -10: a1:=a1+'Å';
  -9: a1:=a1+'nm';
  -6: a1:=a1+'um';
  -3: a1:=a1+'mm';
end;
printh(fvar,'\Scan size: '+a1);
str(offsetX:0:1,a1);
printh(fvar,'\X offset: '+a1+' nm');
str(offsetY:0:1,a1);
printh(fvar,'\Y offset: '+a1+' nm');
if (ldir=TRACE) then
  printh(fvar,'\Line direction: Trace')
else
  printh(fvar,'\Line direction: Retrace');
printh(fvar,'\Rotate Ang.: 0');
str(nx,a1);
str(ny,a2);
printh(fvar,'\Samps/line: '+a1+' '+a2);
str(ny/(dt/100.0):0:3,a1);  { 1/s }
printh(fvar,'\Scan rate: '+a1+' 1/s');
edt:=50000.0/(1.0*nx)/(1.0*ny)*(1.0*dt);
str(edt:0:3,a1); { 10 æs }
printh(fvar,'\Sample period: '+a1+' *0.1 us');
str(igain,a1);
printh(fvar,'\Int. gain: '+a1);
str(pgain,a1);
printh(fvar,'\Prop. gain: '+a1);
str(filter,a1);
printh(fvar,'\Filter: '+a1+' Hz');
if not(dualtrace) then
  printh(fvar,'\Scope dualtrace: Single')
else
  printh(fvar,'\Scope dualtrace: Dual');
str(VerstZ:0:3,a1);
printh(fvar,'\Z sens: '+a1);
printh(fvar,'\ZPiezo type: '+zptype+'type');
str(EichZ:0:3,a1);
printh(fvar,'\Z scale: '+a1);
printh(fvar,'\XYPiezo type: '+xyptype+'type');
str(EichX:0:3,a1);
printh(fvar,'\X scale: '+a1);
str(EichY:0:3,a1);
printh(fvar,'\Y scale: '+a1);
printh(fvar,'\Start context: '+ctx);
printh(fvar,'\Id: '+id);
case mode of
  CCM: a1:='CCM';
  CHM: a1:='CHM';
end;
printh(fvar,'\Mode: '+a1);
if (ctx='STM') then begin
  str(u_tunnel:0:3,a1);
  printh(fvar,'\Bias: '+a1+' mV');
  str(ut_freq:0:3,a1);
  printh(fvar,'\Biasfreq.: '+a1+' Hz');
  str(i_tunnel:0:3,a1);
```

```
      printh(fvar,'\Current: '+a1+' nA');
    end;
    if (ctx='AFM') then begin
      springk:=u_tunnel;
      str(springk,a1);
      printg(fvar,'\SpringK: ',+a1+'N/m');
    end;

    {--- Controller list ------------------------------------}
    printh(fvar,'\*Controller list');
    printh(fvar,'\CId: '+cid);

    {--- Environment ----------------------------------------}
    printh(fvar,'\*Environment');
    str(temperature:0:3,a1);
    printh(fvar,'\Temperature: '+a1+' K');
    str(pressure:0:3,a1);
    printh(fvar,'\Pressure: '+a1+' mbar');

    {--- Image info -----------------------------------------}
    SetIFWHeader:=OK;
  end;
end;

function CloseIFWFile(var fvar:ifwfilestruct):integer;
var ft:text;
     f:file;
    t1:longint;
    tmp:byte;
    i:integer;
begin
  if not(fvar.opened) then
    CloseIFWFile:=FNOPEN
  else if (fvar.hlcount=0) then
    CloseIFWFile:=NHEAD
  else if (fvar.dcount=0) then
    CloseIFWFile:=NDATA
  else begin
    { writing header to file }
    printh(fvar,chr(26)); { ^Z }
    assign(ft,fvar.fname);
    rewrite(ft);
    for i:=1 to fvar.hlcount do writeln(ft,fvar.header[i]);
    close(ft);
    { filling header to 8 kByte }
    assign(f,fvar.fname);
    reset(f,1);
    t1:=filesize(f);
    seek(f,filesize(f));
    blockwrite(f,(@tmp)^,8192-t1);
    { now writing data }
    for i:=1 to fvar.dcount do fvar.data[i]^.WriteToFile(f);
    { WriteToFile schreibt alle Daten Zeilenweise im Integer-Format
      in die Datei f (niederwertiges Byte zuerst) }
    close(f);
    fvar.opened:=FALSE;
    CloseIFWFile:=OK;
  end;
end;

begin
end.
```

Die Funktionalität der TArray-Objekte kann der array-Unit (© Nils Schmeißer) entnommen werden. Die Aufrufreihenfolge

```
  NewIFWFile       Erzeugen einer neuen Datei
  OpenIFWFile      Öffnen der Datei
  SetIFWHeader     Setzen von \*File list, \*Controler list und
                   \*Environment
```

```
  AddIFWData       Hinzufügen von Daten
   ...
  AddIFWData
  CloseIFWFile     Schreiben und Schließen der Datei
```

muß eingehalten werden. Beispiel:

```
uses ifwfile,arrays;

var time:timeinfo;
    fvar:ifwfilestruct;
    p1,p2:PArray;

begin
  p1:=new(PIntArray,Create(60,60));
  p2:=new(PIntArray,Create(1,64));
  with time.start do begin
    hour:=11;
    minute:=20;
    second:=0;
    second100:=0;
    day:=17;
    month:=10;
    year:=1995;
  end;
  with time.stop do begin
    hour:=11;
    minute:=22;
    second:=0;
    second100:=0;
    day:=17;
    month:=10;
    year:=1995;
  end;

  NewIFWFile(fvar);
  OpenIFWFile('test.ifw',fvar);
  SetIFWHeader(time,'textfield','historyfield',
               128,128,0,0,10,-10,TRACE,CCM,
               3200,3300,0,FALSE,
               0.75,80,1,
               'A','C',100,20,30,
               100,
               'STM','LT ITC4',
               'Lehmann control',
               273.15,103.25,
               fvar));

  AddIFWData(p1,TOP,1.23,0.0,'Å',
             'data topography',
             0,0,0,
             0,0,
             0,0,0,0,0,0,0,0,0,
             fvar);
  AddIFWData(p2,SPE,5.78,0.0,'mA',
             'data i-u kennlinie',
             { for Potentiometry }
             0,0,0,
             { Harmonics }
             0,0,
             { Spectroscopy }
             I_U,-5,5,2,0,100,20,30,1,
             fvar);

  CloseIFWFile(fvar);

  dispose(p1,Delete);
  dispose(p2,Delete);
end.
```

4. Header-Beispiel

```
\*File list
\Date: 11:20: 0 17.10.1995
\Data length: 8192
\Text: textfield
\History: historyfield
\*Microscope list
\Scan size: 10.0 Å
\X offset: 0.0 nm
\Y offset: 0.0 nm
\Line direction: Trace
\Rotate Ang.: 0
\Samps/line: 128 128
\Scan rate: 1.067 1/s
\Sample period: 36621.094 *0.1 us
\Int. gain: 3200
\Prop. gain: 3300
\Filter: None
\Scope dualtrace: Single
\Z sens: 100.000
\ZPiezo type: Atype
\Z scale: 30.000
\XYPiezo type: Ctype
\X scale: 100.000
\Y scale: 20.000
\Start context: STM
\Id: LT ITC4
\Mode: CCM
\Bias: 0.750 mV
\Biasfreq.: 80.000 Hz
\Current: 1.000 nA
\*Controller list
\CId: Lehmann control
\*Environment
\Temperature: 273.150 K
\Pressure: 103.250 mbar
\*Image info
\Image: 1
\Doffset: 0
\Type: TOP
\Z scaling: 1.230 Å
\Z offset: 0.000 Å
\Text: data topography
\*Image info
\Image: 2
\Doffset: 7200
\Type: SPE
\Z scaling: 5.780 mA
\Z offset: 0.000 mA
\Text: data i-u kennlinie
\SType: I_U
\USmin: -5.000 mV
\USmax: 5.000 mV
\U0: 2.000 mV
\I0: 0.000 mV
\Numsamples: 100
\Point: 20 30
\Repeat: 1
```