28.0 GROENEVELD, E.: Zur Datenfernübertragung zwischen Mikro-Computer und Zentralrechner

Einleitung

Microcomputer finden einen immer breiteren Einsatz in der Praxis, wobei sowohl neue Einsatzgebiete erschlossen als auch möglicherweise angestammte Aufgaben der Rechenzentren übernommen werden. Eine bestimmendes Characteristikum der Situation ist die Atomisierung, die neben den Vorteilen der dezentralen EDV-Leistung - also dem unmittelbaren Verfügbarwerden für den Benutzer natürlich auch eine ganze Reihe von Problemen mit sich bringt. Über die Vorteile der dezentralen EDV soll hier nicht berichtet werden, sondern über bestimmte Aspekte der daraus resultierenden Probleme und wie diese Probleme in adaequater Weise behandelt werden können.

Im Folgenden wird von einer Situation ausgegangen, in der periphere Aufgaben wie etwa die Produktionskontrolle eines Mastbetriebes oder die Indexberechnung von Zuchttieren erledigt werden müssen. Hierbei handelt es sich im Prinzip immer um Dateneingabe, Datenverwaltung und um Auswertungen, die auf dieser Datengrundlage erfolgen. Die einzige Voraussetzung für die folgende Diskussion ist die Forderung, die anstehenden Aufgaben mit größtmöglicher Effizienz zu lösen. Somit gibt es a priori keine Festlegung ob etwa Auswertungen dezentral oder zentral erfolgen sollen.

Die Kommunikationssoftware

Ein Telekommunikationsnetz in diesem Sinne hätte also dezentrale Aufgaben wahrzunehmen, für die als Arbeitsbasis eine Teilmenge der Daten benötigt werden wie sie auch zentral zur Verfügung stehen müssen. Daraus ergibt sich die Notwendigkeit, Daten - hier jetzt im Sinne von numerischen Werten - von der Peripherie zur Zentrale zu schicken. Neben dem reinen Transportproblem, also der Übermittlung der Daten, müssen noch weitere Informationen zur Zentrale geliefert werden, nämlich um welche Art von Daten es sich handelt und was mit ihnen geschehen soll. In der Zentrale ist der Prozess der Kommunikation ja nicht mit dem Empfang der Daten abgeschlossen. In der Regel müssen die so empfangenen Daten nach einer Prüfung in die entsprechenden Dateien oder Datenbanken geladen werden. Möglicherweise schließen sich dann noch Auswertungen an.

Ein solcher Ablauf kann auf verschiedene Weise realisiert werden. So kann etwa die Datenart in den Daten selbst stehen: zum Beispiel das erste Zeichen eines jeden Satzes bezeichnet die Datenart. Während der Datenübertragung zum Zentralrechner werden die Daten laufend in eine Eingangsdatei geschrieben. Nach der Übertragung wird ein Reihe von Programmen gestartet, von denen jedes eine Datenart verarbeitet. Die Programme lesen die Eingangsdatei und verarbeiten jeweils die Datenart für die sie zuständig sind, wobei das Problem der Behandlung der bereits

abgearbeiteten Sätze behandelt werden muß. Die Ergebnisse und auch Fehlerlisten werden auf dem zentralen Drucker ausgegeben. Diese Art der Organisation ist im Bundeshybridzuchtprogramm (BHZP) realisiert und befindet sich seit 1980 im täglichen Einsatz (GROENEVELD 1985). Die Datenerfassung wird bei diesem System über programmierte Halbleiterspeichergeräte MICRONIC 445 durchgeführt. Die Programme sind in EPROM's abgelegt, sodaß Änderungen in der Programmierung umständlich sind.

Bei einer Revision des Kommunikationskanals für den Anschluß peripherer Systeme im Marienseer Herdbuch Informations System, MAHIS (GROENEVELD 1977, 1978, 1981a, 1981b, 1982), und in der Vermehrungsstufe des BHZP wurde eine größere Flexibilität bei insgesamt geringeren Aufwand angestrebt. Dies ist im wesentlichen die Forderung nach einem offenen System, welches den sukzessiven Ausbau zu immer umfassenderen Lösungen gestattet. Es kommen dort Microcomputer mit dem Betriebssystem CP/M zum Einsatz (CP/M ist ein Warenzeichen von DIGITAL RESEARCH). Durch die Nutzung von Standardsystemfunktion wurde der gesamte Ablauf wesentlich einfacher und effizienter gestaltet als es für die Basiszucht im BHZP möglich war: Vor der Datenübertragung wird auf dem peripheren Rechner ein Stapelpaket aufgebaut, wie es für den Stapelbetrieb des Zentralrechners benötigt wird. Diese Datei enthält somit alle Informationen die zum Ablauf des Programmes auf dem Zentralrechner benötigt werden. In der Tabelle 1 ist ein solches Beispiel angeführt.

Tabelle 1 Stapelpaket zum Zentralrechner

$$STPL$$4711,4711	-	Marke
§ENTER,PERIPHERIE,,55,5-		Start des Stapels
§LADE-1,4711	-	Aufruf des Programms,Stationskennung
1233445454556	-	Es folgen Daten
...	-	"
§REPORT-1,4711	-	Ein neues Programm,Stationskennung
§REPORT-2,4711	-	Ein neues Programm,Stationskennung
§LOGOUT	-	Abschluß des Stapels

Im folgenden Schritt erfolgt die Übertragung dieses Pakets an den Zentralrechner. Die Tabelle 2 gibt schematisch den Ablauf der Kommunikation zwischen Peripherie und Zentralrechner wieder.

Oberhalb des Transportniveus erkennt die Kommunikationssoftware durch die Marke '$$STPL$$', daß im folgenden Daten als Stapelpaket ankommen. Entsprechend werden diese auf eine Datei geschrieben. Sollte die Übertragung nicht korrekt abgeschlossen werden, wird diese Datei gelöscht. Ansonsten wird sie in die Stapelschlange des Betriebssystems eingereiht. Damit ist der Wirkungsbereich der Kommunikation verlassen, die Standardfunktionen des Betriebssystems übernehmen die Resourcenverwaltung. Als spezielle Systemfunktion wird lediglich ein Programmaufruf benötigt, der das Einstellen einer Datei in die Stapelschlange gestattet.

Es muß jetzt noch sichergestellt werden, daß die Ausgabe wieder an die rufende Station zurückgeleitet wird. Dazu wird wie in Tabelle 1 zu sehen ist

die Kennung der peripheren Station an den Programmnamen angehängt. In diesem Beispiel kommt der Stapel also von der peripheren Station '4711'. Von den aufgerufenen Programmen wird diese Kennung aufgenommen. Über ein Unterprogramm wird die vom Programm produzierte Ausgabe hinten an die Datei 'L4711' angehängt.

Tabelle 2 Schematischer Ablauf der Kommunikation zwischen peripherem Rechner und Zentralrechner

Daten von der Peripherie zum Zentralrechner

Peripherie		Zentralrechner
1. Stapel aufbauen - zu übertragende Daten aus der Datenbank extrahieren, blocken und auf eine Ausgabedatei schreiben.		
2. Verbindung zum Zentralrechner herstellen.	LSV2	
3. Stapel übertragen.	---------->	'$$STPL$$4711' erkennen, -> Stapel auf Datei schreiben. Datei in die Stapelschlange einreihen. Stapel zu gegebener Zeit abarbeiten. Ergebnisse an Datei L4711 anhängen.
4. Verbindung abbauen.		

Somit lassen sich von der Peripherie alle auf dem Zentralrechner vorhandenen Programme aufrufen. Die einzige Voraussetzung ist der Einbau der Routine, die die Ausgabe auf die entsprechende Datei umleitet. Aus dem oben gesagten ergibt sich, daß die Ausgabe für eine periphere Station in einer Datei gesammelt wird. Der Ablauf der Übertragung von der Zentrale zur Peripherie ist in der Tabelle 3 dargestellt.

Der Anstoß zur Übertragung erfolgt auch hier von der Peripherie. Statt der Marke '$$STPL$$4711' wird hier '$$HOLE$$4711' als Einleitung des Datentransfers an den Zentralrechner geschickt. Dieser versucht jetzt, die Datei 'L4711' zu öffnen und zu übertragen. Gelingt dies nicht, etwa weil sie noch gar nicht existiert oder aber weil sie gerade beschrieben wird, so wird statt ihrer eine entsprechende Meldung zu Peripherie geschickt.

Auf dem peripheren Rechner werden die Daten während der Übertragung in eine Datei geschrieben. Nach der Übertragung erfolgt die Analyse dieser Datei.

Neben Druckausgaben sollten über einen Kommunikationskanal alle Informationen im weitesten Sinn übertragen werden können, die zwischen einem Zentralrechner und peripheren Systemen ausgetauscht werden müssen. Dazu gehören Daten, die auf dem peripheren System von Programmen benötigt werden, also z.B. die Identifikationen von zur Zucht

zugelassenen Ebern. Diese Daten müssen nach der Datenübertragung in periphere Datenbestände aufgenommen werden, es muß also nach der Datenübertragung ein Prozess gestartet werden, der diese Aufgabe ausführt.

Tabelle 3 Schematischer Ablauf der Kommunikation zwischen peripherem Rechner und Zentralrechner

Ergebnisse vom Zentralrechner abholen

Peripherie		Zentralrechner
1. Verbindung zum Zentralrechner herstellen.	LSV2	
2. '$$HOLE$$4711' übertragen	---------->	'$$HOLE$$4711' erkennen, -> Datei 'L4711' öffnen, Inhalt übertragen.
3. Antwort abholen und auf Datei schreiben.	<----------	
4. Verbindung abbauen.		
5. Entblocken und verteilen: wenn '$$FIL$$Dateiname' -> das folgende bis '$$EOF$$' auf Datei 'Dateiname' schreiben. wenn '$$XEQ$$DO.CMD' -> das folgende bis '$$EOF$$' auf Datei 'DO.CMD' schreiben. sonst -> ausdrucken.		
6. dBASE II Programm 'INIT' starten INIT versucht das Programm 'DO.CMD' auszuführen: Wenn Datei 'DO.CMD' existiert - führe 'DO.CMD' aus - lösche 'DO.CMD' Ende		

Weiterhin sollte es die Möglichkeit geben, neue Programmversionen auf den peripheren Rechnern zu installieren.

Schließlich wäre es wünschenswert, von der Zentrale aus eine vollständige Kontrolle über den peripheren Rechner zu haben, das heißt, beliebige Programme dort zu initieren, Modifikationen an Datenstrukturen und an Daten selber vorzunehmen.

Entsprechend der verschiedenen Arten von Informationen, die zur Peripherie geschickt werden, sind in den Datenstrom Marken eingefügt, die nach dem Transport der Daten zu Peripherie analysiert werden. Die Tabelle 4 gibt ein Beispiel eines solchen Datenstroms. Dabei handelt es sich um das Ergebnis des zuvor übertragenen Paketes aus Tabelle 1. Zusätzlich hat der zentrale Netzverwalter ein direkt nach der Datenübertragung auf dem peripheren Rechner zu laufendes Programm und die neue Version eines BASIC Programms in die Ausgabedatei eingestellt. Die Marke '$$FIL$$Dateiname' im Datenstrom bewirkt, daß die folgenden Daten bis zur Marke '$$EOF$$' auf die Datei 'Dateiname' geschrieben wird. Auf diese Weise können neue Programme oder neue Programmversionen installiert werden. Auch Daten können so auf eine Datei auf dem

peripheren Rechner geschrieben werden.

Die Marke '$$XEQ$$DO.CMD' leitet einen Programmcode ein, der im Anschluß an die Datenanalyse ausgeführt werden soll. An dieser Stelle sollte gesagt werden, daß die Programmierarbeiten auf den peripheren Systemen in dBASEII ausgeführt sind (dBASEII ist ein Warenzeichen von ASHTON TATE). Neben der Tatsache, daß es sich hierbei um ein - für Microcomputersysteme - mächtiges relationales Datenbanksystem handelt, ist in diesem Zusammenhang von Bedeutung, daß dBASEII symbolischen Code ausführt. Dieser wird also in die Datei DO.CMD geschrieben. Anschließend versucht das System (alle Programme auf dem peripheren System inclusive der Kommunikationen werden aus dBASEII heraus gestartet) die Datei DO.CMD auszuführen.

Tabelle 4 Muster einer Ausgabedatei für die Peripherie

```
1                    *************
                     *  LADE-1   *
                     *************
                  17.04.84 12.12.12
        Fehlerstatistik
        ---------------
        Eingelesen  :  4
        Korrekt     :  4
        Fehler      :  0
1                    *************
                     *  REPORT1  *
                     *************
                  17.04.84 12.12.12
        Es sind zur Zeit 132 Eber selektiert.
        Die Selektionsdifferenz beträgt 23 Indexpunkte.
1                    *************
                     *  REPORT2  *
                     *************
                  17.04.84 12.12.12
        Bis jetzt sind 1233 Eber auf Auktionen
        getestet worden. Die mittleren Leistungen sind:
           Testgewicht                      : 129 kg
           mittlere korrigierte Speckdicke: 13,6 mm
           Lebenstagszunahme                :  634 g
$$XEQ$$DO.CMD
ERASE
STORE 0 TO COUNT
DO WHILE COUNT < 20
   STORE COUNT+1 TO COUNT
   § COUNT,10 SAY "Dies ist ein dBASE II  Programm."
ENDDO
RETURN
END
$$EOF$$
$$FIL$$FILE.BAS
100 REM BASIC-PROGRAMMVERSION VOM 14.04.1984
200 FOR I = 1 TO 20
300 PRINT "HALLO, DIES IST EINE NEUE PROGRAMMVERSION."
400 NEXT I
500 END
$$EOF$$
      ****************************************
      *    Neue Programme sind angekommen.   *
      *    Neue Sicherheitskopie anfertigen  *
      ****************************************
```

Nach der Übertragung würde auf dem peripheren Rechner folgendes passieren (siehe auch Punkt 5 der Tabelle 3):

1. Die Ergebnisse der Programme 'LADE-1', 'REPORT1' und 'REPORT2' werden auf dem Drucker ausgegeben.
2. Der hinter '$$XEQ$$DO.CMD' folgende Code, also ERASE..... wird bis zum '$$EOF$$' auf die Datei 'DO.CMD' geschrieben. Diese Datei wird offen gehalten, um eventuell weiteren Programmcode hinten anzuhängen.
3. Der hinter '$$FIL$$FILE.BAS' folgende Text wird auf die Datei 'FILE.BAS' geschrieben. Sollte diese Datei bereits bestehen, wird der alte Inhalt überschrieben. Damit ist eine neue Softwareversion installiert.
4. Der hinter '$$EOF$$' folgende Text wird wieder ausgedruckt, damit der periphere Benutzer weiß, daß er wegen geänderter Programme eine neue Sicherheitskopie anlegen muß.
5. Es folgt die Ausführung von Punkt 6, Tabelle 3. dBASEII versucht, die Prozedur 'DO.CMD' zu starten. In diesem Beispiel wird der Bildschirm mit dem Text 'Dies ist ein dBASEII Programm.' vollgeschrieben. Es dürfte klar sein, daß auf ähnliche Weise jedes von der Zentrale geladene Programm auf dieselbe Art auf dem peripheren Rechner ausgeführt werden kann.

Somit können im Prinzip beide Rechner als entfernte Stapelstationen betrieben werden, wobei die Möglichkeiten auf dem peripheren Rechner einen von der Zentrale initierten Dialog einbeziehen.

Da auf der Transportebene die BSC Prozedur LSV2 eingesetzt wird, ist die Übertragung auf Code aus dem ASCII-Bereich beschränkt. Somit können ausführbare Programme über diesen Kanal nicht zur Peripherie übertragen werden. Da bis auf das Kommunikationsprogramm nur interpretierende Systeme (dBASEII und BASIC) eingesetzt werden, ist diese Restriktion nicht sehr gravierend. Wollte man auch diese Möglichkeit haben, müßte das Transportmodul geändert werden.

Ein wesentlicher Vorteil des im Frühjahr 1984 in Betrieb genommenen Systems liegt in dem völlig automatischen Betrieb in der Zentrale. Auf der Systemkonsole werden bei hereinkommenden und herausgehenden Daten die Stationskennung und die Uhrzeit mitgeschrieben. Auch der weitere Ablauf, das Prüfen der Daten, das Laden in die zentrale Datenbank, der Ablauf von Auswertungsprogrammen erfolgt vollautomatisch. Somit wird der Rechenbetrieb von den peripheren Stationen aus auch außerhalb der normalen Dienstzeit aufrecht erhalten, wenn kein Operateur den zentralen Rechner bedient.

Das Problem des Anschlusses von Rechnern an das Kommunikationsnetz

Auf dem Transportniveu wurde die BSC Prozedur LSV2 realisiert. Sie folgt der DIN 66019. In standard FORTRAN 4 geschrieben gewährleistet sie weitestgehende Portabilität. So läuft sie inzwischen auf 3 verschiedenen Typen von Microcomputerhardware und auf zwei Zentralrechnern. Es sind 4 verschiedene Betriebssysteme und 3 verschiedene Compiler beteiligt.

Eine Voraussetzung für den Einsatz der LSV2 Prozedur ist die freie Programmierbarkeit der I/O Ports und die Gewähr, daß alle einkommenden Daten mit hinreichender Geschindigkeit verarbeitet werden können. Weiterhin müssen die an das Kommunikationsnetz der Bundespost anzuschließenden

Geräte vom Fernmeldetechnischen Zentralamt (FTZ) zugelassen werden. All diese Probleme lassen sich in der Regel leichter bei Microcomputern lösen. Aus diesem Grunde wurde ein CP/M Microcomputer als Pufferrechner zwischen zentralem Rechner und dessen Modem geschaltet. Der eine Port dient als Ausgang zum Modem, während der andere direkt mit dem Zentralrechner gekoppelt ist. Die Aufgabe des Micro's besteht nun darin, die von der Peripherie eingehenden Daten in einem Puffer zu speichern und sobald dies möglich ist, an den Zentralrechner weiterzugeben. Zwischen Micro und Zentralrechner wurde ein einfaches Protokoll implementiert. In diesem Fall wird jedes zum Zentralrechner gesendete Zeichen von ihr als Quittung zurückgeschickt. Erst wenn dies vom Pufferrechner empfangen ist wird das nächste Zeichen gesendet. Wenn also einmal für kurze Zeit auf dem Zentralrechner kein Zeichen abgenommen werden kann, weil sie gerade eine nicht unterbrechbare Sequenz ausführt, füllt sich der Puffer für die einkommenden Daten auf dem Micro. Dieser wird erst wieder geleert, wenn der Zentralrechner wieder in der Lage ist, Daten abzunehmen. Da zum Zentralrechner mit einer Geschwindigkeit von 9600 Baud übertragen wird, die Daten von der Peripherie zur Zeit mit 300 Baud eintreffen, verlängert sich die Kosten verursachende Übertragungszeit nicht.

Allerdings muß der Pufferrechner neben seiner eigentlichen Aufgabe, Daten in beide Richtungen weiterzureichen, noch die Funktionen der Modembedienung durchführen. Hierzu gehört, das Ende der Übertragung zu erkennen und am Modem entsprechende Signale zu setzen.

Bedienung einer großen Zahl peripherer Stationen

Sind von einem Zentralrechner eine große Zahl peripherer Stationen zu bedienen, so kann es zu Engpässen kommen, wenn z.B. am Nachmittag nach getaner Erfassung jeder die Daten zur Zentrale übertragen, Auswertungen in Auftrag geben oder aber Ergebnisse abholen will. Während die Systembelastung über die Stapelverwaltung des zentralen Systems in Grenzen gehalten wird, wird der zentrale Telephonanschluß zu den Stoßzeiten überlastet sein. Eine Vermehrung der Zahl der Anschlüsse und natürlich auch der I/O Ports ist sicherlich nicht effizienteste Methode. Da die Zahl der peripheren Stationen sowohl in BHZP als auch in MAHIS steigt, ist ein Konzept entwickelt worden, das eine automatische Bedienung der peripheren Rechner durch den zentralen vorsieht. Seit Anfang des Jahres 1984 stellt die Deutsche Bundespost Baugruppenmodems mit der Bezeichnung MDB 1200 zur Verfügung. Diese Modems erlauben eine automatische Anwahl von Gegenstationen. Weiterhin arbeiten die Modems mit einer Geschwindigkeit von 1200 Baud. Diesen Vorteilen und den geringen monatlichen Kosten von rund 20 DM steht der Nachteil der Zulassungspflicht durch das FTZ gegenüber. Ist diese Hürde genommen so eröffnen sich allerdings interessante Möglichkeiten:

An der Peripherie wird am Ende des Arbeitstages das Kommunikationsprogramm gestartet, der Rechner an den Modem angeschlossen. Zu gegebener Zeit, also etwa nach dem großen Ansturm auf das Telephonnetz in den Abendstunden um 23 Uhr, beginnt der zentrale Rechner, eine periphere Station nach der anderen anzurufen und deren Übertragungswünsche in Empfang zu nehmen. Nach der ersten Übertragung

beginnt parallel zu den weiteren bereits die Verarbeitung der Daten aus der Stapelschlange heraus. Nachdem die Stapelschlange abgearbeitet ist, die Ergebnisse also zur Verteilung an die peripheren Stationen vorliegen, werden diese wiederum angewählt und die Daten dorthin zurückgeschickt.

Somit kann ein großer Teil der Kommunikation zur Zeit des günstigsten Telephontarifs abgewickelt werden. Außerdem werden Warteschlangen vor dem Kommunikationsanschluß am Zentralrechner vermieden. Eine weitere Folge dieser Organisation ist, daß die Telephonkosten bei der Zentrale anfallen, vielleicht nicht in allen Fällen ein wünschenswerter Nebeneffekt.

Das in umfangreicheren Kommunikationsnetzen nicht unerhebliche Problem der Softwarewartung wird auf eine unerhebliche Größe reduziert: neue Softwareversionen brauchen lediglich in die für jede periphere Station vorgesehene Ausgabedatei eingestellt zu werden. Dies geschieht in der beschriebenen Situation über ein Programm welches als Eingabe die zu verteilende Datei einliest und auf alle Ausgabedateien verteilt. Bei der nächsten Ausgabeanforderung seitens der peripheren Stationen wird die neue Software mit übertragen und automatisch installiert. Somit ist sichergestellt, daß zum erstmöglichen Zeitpunkt die neue Software benutzt wird. Der Aufwand auf seiten der Zentrale liegt (neben der Erstellung der neuen Version) im Aufruf des Verteilungsprogramms und der Eingabe der zu verteilenden Datei. Dabei ist unerheblich wieviele Stationen im Netz arbeiten, wie deren Hardware aussieht, oder sogar welches Betriebssystem sie fahren. Die dafür notwendige Anpassung wird einmal bei der Implementierung der Kommunikationssoftware gemacht.

Neben den Schnelligkeit mit der ein neuer Softwarelevel verteilt werden kann ist vor allen Dingen der geringe organisatorische Aufwand hervorzuheben, da dieser im Gegensatz zu anderen Verfahren (etwa das Versenden von Disketten) unabhängig von der Zahl der peripheren Stationen ist.

Zusammenfassung

Das vorgestellte und seit Anfang 1984 in der Praxis eingesetzte System eines offenen Kommunikationsnetzes zwischen peripheren Mikrocomputern und Zentralrechner basiert auf der Stapelverarbeitung von Daten. Dadurch werden sowohl die Übertragungskosten gegenüber dem Dialogbetrieb minimiert als auch Ein/Ausgabekanäle optimal ausgenutzt. Weiterhin wird die Systembelastung über die Standardfunktion der Stapelverwaltung reguliert. Der Ablauf an der Zentrale wird vollautomatisch ohne jede Operateurbedienung durchgeführt. Die Softwarewartung der peripheren Systeme wird von der Zentrale initiiert und über den Kommunikationskanal abgewickelt. Die Installation neuer Softwarekomponenten wird automatisch bei der ersten Kommunikation zwischen peripherem System und Zentralrechner vorgenommen, wobei der Aufwand unabhängig von der Zahl der peripheren Systeme im Netz ist. Durch den modularen Aufbau können mit geringen Aufwand alle auf dem Zentralrechner vorhandenen Stapelprogramme für die peripheren Anwender nutzbar gemacht werden. Somit können auch neue Anwendungen leicht im Gesamtsystem implementiert werden.

Literatur

Groeneveld, E., 1977. The use of programmable desk calculators for estimating breeding values on auction sales for breeding pigs. 28 th Annual Meeting of the EAAP, Brussels.

Groeneveld, E., 1978. Neue Entwicklungen in der elektronischen Erfassung und Verarbeitung von Felddaten. Tierzüchter 5: 192-194.

Groeneveld, E., 1981a. Sammlung und Kombination von Stations- und Felddaten als Voraussetzung zur züchterischen Betreuung von Schweinezuchtpopulationen. Landbauforschung Völkenrode 2: 72-75.

Groeneveld, E., 1981b. Use of micro computers in the field for estimating breeding values: a system to automatically collect all data handled on a population in a computer compatible way for further processing. 32 nd Annual Meeting of the EAAP, Zagreb.

Groeneveld, E., 1982. Mariensee Herdbook Information System, a tool for genetical population control. II. World Congress on genetics applied to Livestock production. Madrid, 4.-8. October 1982. Vol. VIII. 6c-19, p. 584-592

Groeneveld, E., 1985. Integrated data acquisition, transmission and processing systems in two pig breeding schemes in the Federal Republic of Germany Livest. Prod. Sci. im Druck

DATENERFASSUNG IM FELD (HAND-HELD-COMPUTER)