

CONVERTING DATE & TIME DATA FROM CSI ARRAY DATA FORMAT FOR USE IN SPREADSHEET & DATABASE SOFTWARE

1. INTRODUCTION

Many Campbell Scientific Dataloggers store date and time information as Julian date (number of days into the current year) and HHMM (where HH=hours and MM=minutes) formats to minimise storage space and maximise readability of the raw data when viewed as text files.

Microsoft's database (MS Access) and spreadsheet (MS Excel) programs store dates and times as real numbers, where the integer portion of the number represents the number of days counted from some base date (usually January 1, 1900), and the fractional portion represents the time of day. For example, June 1, 2000 at 10:00 a.m. would be stored as '36678.41667'.

This note describes the functions and steps needed to read in the data files and convert the date and time formats output from Campbell Scientific dataloggers to the formats used in Microsoft Access and Excel software. Most other leading software database and spreadsheet packages operate in a similar way.

2. IMPORTING CS DATA FILES – GENERAL PRINCIPLES

Software that supports collection of data from dataloggers that store 'array data' offer two or three formats in which they can write data to disk. The option for 'comma separated' (often known as CSV) is the best to use when you intend to read data into another package. If data has been collected in raw binary format or 'Printable ASCII' it is often simpler to use Campbell Scientific's Split program to reformat the data to CSV, rather than trying to decode and import the data in these alternative formats.

Split can also be used to sort and extract specific data arrays from a file of mixed array types (see the documentation for Split). With later versions of Split (Windows version 2.0 and later, as shipped with LoggerNet 2.0 and later) you can convert the time stamp values directly into a time value compatible with most other programs (using the Date function and for Excel/Access using the format 'serial'). The sorted and converted data should be saved as 'comma' format with the relevant file extension (see below).

If you have an earlier version of Split, or do not have access to Split, you need to read the comma separated data into the analysis program and use time conversion functions similar to those outlined in sections 3 and 4 below to convert the time values into a serial timestamp that the program will recognise. The following guidelines for reading the data file into the analysis program are relevant whether Split is used or not.

When you save data to disk from Campbell Scientific programs the file will normally be given a '.DAT' or '.PRN' suffix. In some cases, you can greatly simplify transferring the file to that program by specifying a different filename suffix. For many spreadsheets a '.CSV' suffix automates the opening and import of the file without further intervention, although this can cause problems in non-English versions of Windows (see below). For Microsoft Access a '.TXT' suffix is required before any import can be attempted. Please see the manual or help system of the program you intend to use to find the best suffix.

When using older versions of data analysis programs, non-English users should be careful to check data is imported correctly. This is because the CSV format uses a comma (',') as the field separator and point/full stop ('.') to indicate the decimal separator. In many countries this convention is not used for normal data display. The local preferences are set in the Regional Settings in the Windows control panel and are usually set based upon the country specified when Windows was installed. For instance, in many European countries, a ',' is used as the decimal separator when displaying numbers.

Many programs will use the Windows Regional settings when displaying data. When CSV data is read there is a risk that the program may also try to use the Windows specified separator characters and interpret the data incorrectly. As CSV is a defacto standard, most packages will expect the comma field separator. However others may refuse to read the data, may read it but show obvious corruption or even appear to read and display it correctly but where the stored values are corrupt; this may only be apparent during later processing of the imported data. Microsoft Excel 97 is an example of where this problem occurs.

Many recent programs have settings that allow you to specify the delimiters between the data fields and the decimal separator in imported files to the Windows defaults for display. In Excel and Access 2000 onwards, for instance, the separators can be confirmed using the Advanced button of the import Wizard.

If the data has not been pre-processed with Split to separate out the different types of arrays, the 'sort' functions of many packages can be used to arrange and select different arrays for analysis. Complications can arise, though, when reading in such data as different arrays can have different numbers of elements and this can prevent the file being imported correctly, e.g. with Access. In some cases, editing the file to ensure the *first* line of the array has the same number of elements as the longest array present can overcome this problem. It is also often the case that the position of timestamp values may be different in the different arrays making timestamp conversions throughout the file a little more difficult.

If Split has not been used to convert the time format, a new column/field element has to be created in each row/record of data to hold the new time value. That value is then used as the basis for time analysis or graphical display. Examples are given below for formulae that convert the timestamps in Microsoft Access and Excel. Advanced users may be able to define macros or scripts to automate the creation of the new field and then convert the timestamps.

3. READING DATA AND CONVERTING DATE/TIME ELEMENTS TO MS ACCESS DATABASE TABLES

The following sequence shows a typical process of importing a file into Access and converting the time stamp:

1. In Access, open a new or existing Access Database and select File/Get External Data/Import. You may have to change the filename extension to *.TXT and select 'Files of type' as Text. In some versions you can also create a new database from scratch by using the File/Open option.
2. Select the data file and choose 'Import'. A wizard should guide you through the process of choosing commas as the field separators and assigning field names.
3. Click on the new table and then on 'Design' and change the field names for each field, if you haven't done so already, so that they describe the data in the records (e.g. ArrayID, Year, Day, HHMM, Sec, Battery, etc.). Leave the 'Data Types' as Numbers.
4. Switch to Query and create a new query (do not use the 'Query Wizards').

5. Add the new table and drag the fields, one at a time, to the query fields.
6. Insert a new column in the Query Design window (or move to a blank one at the end) and enter the following formula all in one box on the 'Field:' line:

Expr1: CVDDate(([Year]-1900)*365+1+Int((([Year] -1901)/4)+[Day]+Int([HHMM]/100)/24+([HHMM]/100-Int([HHMM]/100))*100/60/24+[Sec]/60/60/24)

('Year', 'Day', 'HHMM' and 'Sec' are the names of those fields in the example table; your specific table may be different. Remember that field names used in formulae must be enclosed in square brackets, e.g., [Year].)

This delivers a numeric expression formatted as a Date/Time expression. The explanation of each part of the formula is as follows:

Section of formula:	Results:
CVDDate	Converts decimal number to date format
([Year] -1900)*365+1+Int((([Year]-1901)/4)	Delivers the number of days since the beginning of the 20 th century through the end of last year taking into account leap years
+ [Day]	Adds the Julian date to the above
+Int([HHMM]/100)/24	Converts the hour portion of the HHMM time field to a fraction of a day
+((([HHMM]/100) - Int([HHMM]/100))*100/60/24	Converts the minutes portion of the HHMM time field to a fraction of a day
+ [Sec]/60/60/24)	Converts the seconds field in to a fraction of a day

If your *.DAT file doesn't include the year you can substitute a fixed number of the current year for the [Year] field. If it doesn't include the HHMM or seconds, just omit those portions of the equation.

4. READING DATA AND CONVERTING DATE/TIME ELEMENTS FROM CSI DATA FILES TO SPREADSHEETS

As discussed above, the initial import of CSV data into many spreadsheets can be simplified by either saving the file directly from the collection software or renaming the file to have a .CSV rather than .DAT suffix. Spreadsheets such as Excel will automatically open such files and read each value into a separate cell in the spreadsheet.

The date and time must then be converted by creating a new column and inserting a formula of the form:

Expr2:=(B1-1900)*365+1+INT((B1-1901)/4)+C1+INT(D1/100)/24+((D1/100)-INT(D1/100))*100/60/24+E1/60/60/24

This formula is based on the equation shown for Access above where a detailed explanation can be found. This example has the year stored in column B, the Julian day in column C, the time as HHMM in column D and seconds in column E. This formula should be copied to as many rows as you have data. If you do not have year values in your data you need to enter the value of the current year instead. If you do not have HHMM and/or seconds data then these terms can be removed.

The resultant values hold a number that is the number of days since 01 Jan 1900 and will typically be a value of approaching 40000 for current data. To see the time in a more standard presentation, in Excel, select the column holding these values and then use Format, Cells and select 'Time' and one of the time formats offered.

5. READING AND CONVERTING FIELD FORMATTED DATA AS OUTPUT BY SPLIT (*.PRN files)

If the original CSV data is not available it is also possible to read and convert field formatted data files output by Split. If the files were output as report files they will need to have any header rows removed to leave just the columns of data, as is found in the Split PRN file format.

MS Excel/Access allows you to import fixed-width fields from ASCII files by choosing FILE/OPEN and clicking on the appropriate column indicators. You can split up the results of the original Split date function (usually 'MM<space>DD') into two fields and once you have defined the fields, you can enter formulas in new columns to convert the year and day fields into dates and the HHMM and second fields into times.

For example, after splitting up the MMDD field into two fields, you can simply use the DATE(year, month, day) function to pick up the data from the relevant fields and convert it to a date. Use the Time(hour, minute, second) function to pick up the relevant fields to create a time, but you will have to enter functions within each parameter to convert from HHMM to separate hour and minute values. The formula looks like:

```
TIME(INT(HHMM/100),(HHMM/100-INT(HHMM/100))*100,second)
```

where *HHMM* is the cell address for the HHMM data
and *second* is the cell address for the seconds data.

If you want one serial time value to work with, you can add the date and time formulae in one cell or add the results from the two cells into a third cell and change that cell or column to any of several numbers or date/time formats.

Acknowledgments

This technical note is based on an Application Note originally written by Bryan Dixon of Campbell Scientific Inc, Logan, Utah, to whom full acknowledgement is made.