ANA

User Guide

Prepared by:

Paul C. Tam

For ANA Version 0.15

Published on 21 December 1990

Printed on 5 June 1991

Paul C. Tam
P.O.Box 390102
Mountain View CA 94039
U.S.A

Table of Contents

1 Highlights 1

2 Introduction 2

3 Inter-operability 3

4 Platforms support 3

5 Conventions 3

6 Summary of commands 4

7 How to start ANA 5

8 User input 5

9 Report 6

10 Structure mapping 8

11 Command Descriptions 12

1 Highlights

Interpret binary data in structure user defined.
Rearrange data bytes before interpretation.
Report current machine data types.
Dump binary data in very flexible format.
Work with multiple files on same screen.
Same user interface across various platforms.
Built in calculator and ASCII/hex/decimal converter.
Save output to disk for future use.
Search for patterns.
Execute operating system specific commands without exiting.
And more......

2 Introduction

ANA is an utility program designed to assist users (especially software developer) who are interested in analyzing the binary contents of a file. This program may be easier for users who know C since the terminology used here is C like.

Its major function is primarily to display the hexadecimal contents of a file interactively. However, what makes ANA unique from most other file dump utilities is perhaps, its ability to not just dumping but ANAlyzing file contents.

Sometimes data files are an array of records, each record contains information of different types. For example, the data file may be a control file of a print queue. There are a number of records in there to represent the number of files waiting to be printed. Each record in turns contains different fields, these fields may indicate the file name to be printed, its priority and so on. They may have data type of character (1 byte), integer (2 or 4 bytes) and ASCII string. Using ANA, user can create an ASCII file in which the structure is defined. ANA then maps the data file into the structure and interprets them as a series of fields instead of a string of bytes.

On top of it, there are a lot of features built in to make this utility more flexible and useful. Some of these features include: able to dump the display buffer into a file, set the display length and base, pack the display and search for combination of bytes. Please read the rest of this document to explore the full functionality of this software, since some of the newly added features are not part of this introduction.

ANA is designed to be highly portable, versions are available at various machines.

Any comments and improvements to this software are welcome, please direct comments to

Paul C. Tam
P.O.Box 390102
Mountain View CA 94039
U.S.A.

3 Inter-operability

ANA is extremely portable, there are versions running on various platforms and various operating systems. All versions have exactly the same user interface.

4 Platforms support

ANA is currently available in the following platforms:
PC DOS
OS/2
HP9000/300 HP/UX
IBM RS6000 AIX 3.1
IBM RT AIX 2.1
IBM PS/2 AIX 1.2
Sun Sparc SunOS 4.03c and 4.1
Sun 3 BSD 4.2

5 Conventions

The following conventions are used in this document:

<text**>**	text inside angle brackets is mandatory.
[text**]**	text inside square brackets is optional.
text **/** text	slash between texts indicates **one only one** of those text can be entered.
text **|** text	vertical bar between texts indicates **one or more** of those text are acceptable.

6 Summary of commands

In order to satisfy some impatient readers who do not like to read the whole document like you and me and everybody, this section summarizes the commands available in this software. To get a better description on these commands, please refer to their corresponding sections.

?	- ANA commands available.
! <command>	- run operating system specific command
ENTER	- display the next report.
a <address>	- set new start address.
b [a/i/d]	- toggle or show address/input/data base.
c <operation>	- calculator functions.
d [file]	- download new Structure Description File.
D	- continuously dump until end of file.
i	- environment information.
l <length>	- set new report length
m [i/l/f/d]	- map data to structure or change alignment.
o [file num]	- open new work file.
p	- toggle packed display mode.
q	- quit analyzer.
s [pattern]	- search for combination of bytes.
t [file]	- transfer data buffer to disk.
u <file id>	- use work file with file id.
V	- display current version.
w <width>	- set display row width
z	- zap old data with new data.
0	- redisplay last report
+ <n>	- skip forward n pages
- <n>	- skip backward n pages

7 How to start ANA

ANA can be invoked in one of the following ways:

1) ANA
2) ANA <data_file_name>

If no argument is supplied, ANA will prompt for a file name.

8 User input

Some ANA commands take arguments, these arguments can be of form hexadecimal, decimal, ASCII character or ASCII string.
For numerical inputs, ANA internally use a default setting to interpret them, this default is called input base and can be manipulated with the **b i** command. Initially all numerical inputs are interpreted as decimal, hence input like 1ff will be an error. If user explicitly set the input base to hexadecimal, 1ff will be an valid input and input like 123 will be hex 123 (291 decimal) rather than 123 decimal. However inputs can be overridden by a prefix. Any numerical inputs prefixed by **0x** are always hex no matter what the current default setting is and any inputs prefixed with **** are always decimal.
For ASCII inputs, single ASCII character must be between single quotes, where ASCII string, can be between any pair of delimiters. e.g. ’string’ and sstrings both represent the word string, only that one is delimited by ’ and the other is by s.

9 Report

When ANA is used for data dumping, data is display in form of a report. Generally this report can be in packed or unpacked format:

Unpacked Hexadecimal (Default) -
0x00000000: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F |................|
0x00000010: 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F |................|

The same data in packed mode -
000102030405060708090A0B0C0D0E0F
101112131415161718191A1B1C1D1E1F

In the standard report, each report line starts with a starting address which indicates the address of the first byte in the line. It then are followed by the file data. On the right hand side there is a string in between two | signs, this string is the ASCII representation of the corresponding data. If the data is not an ASCII displayable character, a period (.) is displayed.

Sometimes it is convenient to pack the data together, this is what the packed format will provide. There will be no address, no ASCII representation and no space between data.

Sometimes data or address make more sense if they are in decimal, even though ANA provide the convertion function to convert hex to decimal, it may be crumblesome to do so. ANA allows users to set to display data and/or address in either hex or decimal.

Sometimes there is a need to display multiple reprots on the same screen, e.g. for comparison. This can be done by setting the report size to a sumall number, hence allow more reports to be display on the same screen. To take this a step further, it will be nice that each report belong to a different data files (this feature will be discussed in section x.x).

When report data is set to decimal, each data is display in 3 digits, the default width in this case will be too long and the report line will wrap around the screen; or may be it makes more sense to display 10 bytes on each report line. Whatever reason there is, it will be nice to be able to set the width of each report line.

When doing analysis on a data file, it is essential to have the ability to jump around different addresses.

The following are the parameters affecting the report format and their initial default settings and the corresponding commands to set them:

Parameters	Defaults	Commands
Pack Mode	Unpacked	p (packed)
Address Base	Hexadecimal	b a (address base)
Data Base	Hexadecimal	b d (data base)
Buffer size	240 bytes	l (length)
Report width	16 bytes	w (width)
Start address	0	a (address)

10 Structure mapping

Mapping structure is a unique feature in this software. Rather than just dumping the data file in bytes, user can define a structure definition file (hereon called SDF) from which the data can be interpreted in a more flexible way. The SDF is a pure ASCII file in which each line represents one data field and the whole file together defines a structure to be mapped.

The way to use this feature is of the following steps:
First, a SDF is created through any editor, if this file is named "ana.fmt" ANA will set up mapping automatically, otherwise user has to load a SDF before mapping is enabled.
Second, display the beginning of the data structure to map.
Finally, activate the mapping command to map the data buffer.
Note that mapping starts at the beginning of the report just display.

Each line in the SDF represents one data type field, every line has the following format:

keyword user_defined_name [length/rearrangement]

The keywords currently supported are:

int	signed integer
char	single character (byte)
string	string of characters
long	long signed integer
short	short signed integer
ulong	unsigned long integer
ushort	unsigned short integer
uint	unsigned integer
float	floating point number
double	double floating point number

The user defined name is used to assist user to identify the field, its content is arbitrary and is limited to 20 characters. Name more than 20 characters will be truncated.

In case of the string data type, the third field is mandatory and contains the number of bytes in the string. For other data type, the third field is optional and if present, contains the rearrangement sequence. For instance, an integer type is 32 bit in a machine (4 bytes), an rearrangement value of 4321 means the 4 bytes are reversed before it is mapped into an integer value. The rearrangement value in this case can be any combination of these 4 digits. This feature is useful if data file is created in one architecture and being interpreted in another architecture (e.g. read a DOS file in a UNIX platform). The number of digits in the rearrangement must be the same as the number of bytes that make up that particular data type, and it must be consecutive numbers begin with 1. Note that rearrangement does not mean anything for a character field.

To allow more flexibility, it is also possible to interactively download a new SDF so that more than one structure can be analyzed in a data file.

ANA assumes architecture that does even boundary alignment, i.e. for data type of even byte, ANA assumes data always start at even address. Therefore, if your architecture allows these even byte data type be started at odd address, the mapping facility is probably not useful. There are not much of this architecture, fortunately. However, for 32 bits data types, many architectures allow even boundary but many align in 4-byte boundary. To solve this problem, ANA allows user to choose between the two through the 'm [i/l/d/f]' command.

EXAMPLES:
Structure mapping is best illustrated through an example. Suppose there is a file of employee records, each record starts with an employee name of 10 characters, then an employee number of type long integer, followed by his salary which is of type integer. Let's further assume that integer is two bytes and a long integer is four bytes. Instead of just dumping the data file in bytes, it is more useful to dump them in a more descriptive form, in this case a string, a long integer and an integer. The SDF will look like this:

```
string   employee_name     10
long     employee_no.      4321
int      employee_salary
```

Suppose we have the SDF named ana.fmt so that mapping will be enabled when ANA starts. When ANA is executed, ANA detects the existence of the SDF, it will then build the internal structure and enable the mapping facility. User can display the portion of data that needs to be analyzed, and then activates the mapping command. It is important to remember that ANA starts analyzing from the beginning of the report.
For example, the report just shown contains:

```
4A 6F 68 6E 20 44 6F 65 20 20 3F 42 0F 00 13 88
```

the display from mapping will be the following:

```
employee_name = (John Doe  )
employee_no. = 999999 (0xF423F)
employee_salary = 5000 (0x1388)
```

Notice that the string is shown inside a pair of parenthesis. The integers are shown in decimal and have their corresponding hexadecimal values. Also, for the employee_no. field since there is a rearrangement value of 4321, ANA will reverse those bytes before interpreting it.

User should be aware that most computer always put data into their corresponding type boundary. For example, for a machine that uses a two byte integer, a structure like

```
structure {
        character A;
        integer 1;
}
may be stored as follow:
        41 XX 00 01
            or
        41 00 01
```

In the first case, the character falls in the even boundary, since the integer has to start also at the even boundary, there is a garbage padding byte in between which does not mean anything. While in the second case, the character falls in the odd boundary and therefore the integer can be put right after the character.

For the same reason, sometimes it is very confusing to just look at the data byte by byte. It is better off to use the structure mapping.

Furthermore, different CPUs have different characteristics. Some align integer and long integer into even boundary while other might align integer in even boundary and long integer in 4 byte boundary. This software will allow users to customize the alignment. Command m [i/l/f/d] specifies the alignment of the data type used in structure mapping.

To make life harder, some CPUs swap bytes (our beloved 80x86 architecture for example) and some don't. The third optional field in the format file are just for that. It specifies the data rearrangement sequence. For example, an integer made of 4 bytes is stored in a file as 0x11 0x22 0x33 0x44 (0x11223344), a line in the format file:

    int sample

yields an output of

    sample = 207454020 (0x11223344)

but if the line is written as:

    int sample 4321

the output will be:

    sample = 1144201745 (0x44332211)

This feature is really useful if for example, sometime tries to dump a file created in a 68000 machine in a 80x86 machine.

11 Command Descriptions

11.1 a <address> ==> Set new starting address

Display report starting at the given address. This command allows user to jump around data file.

11.2 b [a][d][i] ==> Toggle base

In the unpack mode, the address of the first byte of each display line is also shown (see display format on section 8, 9). This address can be of base hexadecimal or decimal, command 'b a' toggles the base.
The data itself can be displayed in either hexadecimal or decimal, command 'b d' toggles the data base.
Numerical input can be entered in either decimal or hexadecimal, command 'b i' set the default input base. Input can always be entered with prefix to override the default.

11.3 c <operation> ==> Calculator/Converter

Sometimes it is necessary to do some simple arithmetic operations on the data displayed. A simple set of arithmetic functions are available in ANA. Currently, the calculator can only do integer arithmetic and is limited to two operand and one operator, i.e. no cascade (with only one exception for conversion which takes one operand and no operator). The following are examples and descriptions of **all** available operations.

```
c X * Y     ( X multiply Y )
c X / Y     ( X divided by Y )
c X + Y     ( X plus Y )
c X - Y     ( X minus Y )
c X % Y     ( reminder of X divided by Y )
c X & Y     ( X bitwise AND with Y)
c X | Y     ( X bitwise OR with Y)
c X ^ Y     ( X XOR Y )
c X > Y     ( X right shift Y bits )
c X < Y     ( X left shift Y bits )
c X         ( give decimal/hex/ASCII representation of X )
```

X and Y here can be an integer, a single ASCII character or a memory. In case of an ASCII character, its ASCII value is used. There are 10 memories stored internally by ANA, they are named m0, m1... to m9. c m0 + m1, for example, will be adding memory 0 to memory 1. It is a memory ring and wraps back to m0 after m9 is used.

11.4 D ==> Continuously dump

The whole work file starting at current location will be displayed continuously until the end of the file.

11.5 d [file name] ==> Download structure description file

Each structure description file maps only one structure, sometimes it is desirable to map data to a different structure. This command loads another description file for the next mapping. If a file name is not entered after the command, the user will be prompted for.

11.6 i ==> Information desk

The following environment information are shown:

- sizes on various data types.
- current work file name and number.
- number of total work files opened.
- maximum work files allowed.
- numeric user input base.
- data display base on report.
- address display base on report.
- mapping alignments for structure mapping.

11.7 l <length> ==> Set new buffer length

Define the size of the data buffer on the next display.

11.8 m [i][l][f][d] ==> Map data to structure

Maps data in the report just displayed into the structure described by the SDF. Mapping currently starts at the beginning of the report buffer, therefore user may have to adjust the starting address before the mapping.
Data type will normally be aligned in a structure. For example, a 'short' after a 'char' will be put in even boundary and the byte after the 'char' is meaningless. ANA allows users to specify its alignment boundary. The arguments are 'i' for int, 'l' for long, 'f' for float and 'd' for double. Their default values are displayed in information desk through command 'i'. (Read section 10 for details)

11.9 o [file number] ==> Open new work file

Open a new work file, the new work file number is provided as the parameter. This number can be the next work file number or can be an existing file number. In case of an existing file number, the existing work file will be replaced and no longer accessible. If no file number is supplied, ANA will assign the next file number automatically.

11.10 p ==> Toggle packed display mode

As discussed in section 8, 9 above, display format can be either packed or unpacked, this command toggle this display format.

11.11 q ==> Quit analyzer

Terminate and exit program.

11.12 s [pattern] ==> Search data

Search for a certain pattern. Pattern can be sequence of number of an ASCII string in between a pair of matching delimiters. Pattern must be specified at the very first time, then in the next search, if no pattern is provided, the previous pattern is search again, otherwise search for the newly supplied pattern.

11.13 t [file] ==> Transfer data buffer to disk

It is possible to store the buffer just displayed into a disk file, using this command will do just that. At the first execution of this command, the user will be prompted for the disk file name unless it is entered with the command. Any subsequent transfer will be appended to the named file and any file name entered in the command line will be ignored.

11.14 u <file number> ==> Use different work file

If there are multiple files opened, this command is used to switch to a different work file.

11.15 V ==> Display current version

This command displays the current version of the software and copyright message.

11.16 w <width> ==> Set display row width

Especially after changing to packed display format from unpacked format, usually it is desirable to display more data in one line. This command allows user to adjust the display width.

11.17 z <data> ==> Zap old data with new data

Replace the old data starting at the beginning of the current display buffer with the supplied new data. Data can be a sequence of numbers or an ASCII string in between a pair of delimiters.

11.18 ? ==> Help
Display a brief description of commands available. This is useful for commands review.

11.19 ! <command> ==> System command
Execute an operating system specific command.

11.20 ENTER ==> Display next buffer
Data is read from file into the data buffer and displayed. Then the next starting address is updated so that the next ENTER will display the following data.

11.21 0 ==> Redisplay buffer
Sometimes data may be scrolled off the screen, this command will redisplay the report that was just displayed.

11.22 + ==> skip forward
Skip over a number of report pages.

11.23 - ==> skip backward
Skip back a number of report pages.

***** END OF DOCUMENT *****