EDOC099

EUROGAM PROJECT

NSF DATA ACQUISITION SYSTEM

Autofill software technical manual

Edition 2.4
December 1994

Nuclear Structure Software Support Group
Liverpool University

1 Introduction

This is the full documentation for the Eurogam Autofill System software, how to install it, an overview of design and coding, and a brief guide to code maintenance.. It is assumed that the reader is familiar with VxWorks, SUN and UNIX and also has an idea on how the Autofill functions as a whole. The code is written in C under SUN.OS (UNIX) using GNU and VxWorks development tools. It was initially developed under OS/9, so some of the design choices reflect this.

This document reflects version 2.4 of the software.

2 Installation

This part of the document describes how to install the Eurogam Autofill software on a new site. The code is portable on object level so recompilation is not neccessary when porting it to another SUN/MVME147/VxWorks setup. The full system will be delivered as a tar-file.

Although certain values (eg path names) may vary between the sites it is recommended that all sites use the same hierarchy of files, directories and stick to the same configuration of the actual Autofill hardware.

2.1 IO configuration

The Autofill software assumes that the jumper-selectable parameters of the IO are set as follows:

2.1.1 XVME560

- SHORT IO operation: J2B inserted, SWITCH BANK 2 - sw8 closed
- SUPERVISOR AND USER MODE ACCESS: SWITCH BANK 2 - sw7 closed
- BASE ADDRESS: 0x0400, 0x0800, ... , 0xf800, 0xfc00
- DIFFERENTIAL INPUT: J10A, J11, J14B inserted
- VOLTAGE RANGE: +- 10 V: J6,J9 inserted, J5,J7,J8 removed
- DIGITAL DATA FORMAT: Two's complement: J3B and J4B inserted

Base address setting (SWITCH bank 2): switches 1 to 6 selects the base address. Examples:

```
0x1000: closed, closed, open,   closed, closed, closed
0x2000: closed, closed, closed, open,   closed, closed
0x4000: closed, closed, closed, closed, open,   closed
0x4400: open,   closed, closed, closed, open,   closed
```

2.1.2 XVME260

- SHORT IO operation: J2B inserted, SWITCH BANK 2 - sw8 closed
- SUPERVISOR AND USER MODE ACCESS: SWITCH BANK 2 - sw7 closed
- BASE ADDRESS: 0x0400, 0x0800, ... , 0xf800, 0xfc00

2.1.3 XVME240

- SHORT IO operation: J2B inserted, SWITCH BANK 2 - sw8 closed
- SUPERVISOR AND USER MODE ACCESS: SWITCH BANK 2 - sw7 closed
- BASE ADDRESS: 0x0400, 0x0800, ... , 0xf800, 0xfc00

2.1.4 MVME712M

The jumpers on the MVME712 must be set to enable those of the serial ports you want to use. The software assumes that all of them are in use and provides login daemons for all four of them.

2.1.5 IO requirement EUROGAM phase 2

Three XVM560 cards at addresses **0x4000**, **0x4400**, **0x4800**.

Three XVM260 cards at addresses **0x0400**, **0x0800**, **0x1000**.

Two XVM240 cards at addresses **0x2000**, **0x2400**.

Note that these addresses are *recommended* as standard and correspond to the configuration which is delivered in the file **egam2.src**. Chosing different addresses means you will have to change the configuration source file.

2.2 FILE ORGANISATION

The Autofill systems depends on several support files residing in four directories. This section will give a brief overview of the files which are required for the system to operate.

2.2.1 BOOT DIRECTORY

This is where the VxWorks related files reside, including VxWorks itself. It also contains the startup script. The name of the directory obviously depends on the chosen boot parameters (fields **filename** and **startup script**). In the setup for Eurogam we have chosen the boot directory to be */eg/boot/* $<$ *cpuname* $>$. This directory contains the following files (a benchmark startup.cmd is supplied on the delivery tar file):

```
-rw-rw-r--  1 startup.cmd
lrwxrwxrwx  1 vxWorks -> ../mv147/vxWorks
lrwxrwxrwx  1 vxWorks.sym -> ../mv147/vxWorks.sym
```

The VxWorks files are links to files appropriate for the CPU. These links must be installed by yourself onto wherever your mv147 VxWorks code resides. Note that the cpu name for the Autofill for phase 2 is **egaf**, so this directory will have the name */eg/boot/egaf* when it is loaded from the tar file.

You will also have to edit parts of **startup.cmd** (server name, inet addresses etc.) but the lines concerning the Autofill path names need not be edited unless you intend to use a non-standard setup. Remember to change the **setRloginHost** and **setRshellHost** entries to something appropriate for your site. These site dependent parameters **must** be set. See STARTUP in the code maintenance section of this document.

2.2.2 WORK DIRECTORY

This directory is */eg/autoFill/work* by default. The files **egam2.src** and **passwd** are supplied on the delivery tar-file and reflect the default configuration name of *egam2*. These files MUST be present for the Autofill to work. In addition some tools to view the state of the Autofill from a remote machine are included.

The files are:

- $< configname >$.src
- passwd
- lcheck
- afview
- afmon
- afdate
- netcontrol

The last five files are not essential to the operation of the Autofill.

The following files will be created as the Autofill is running and are only mentioned here for future reference.

- $< configname >$O.bin
- $< configname >$C.bin
- $< configname >$P.bin
- $pos01.log$
- $posxx.log$
- $pos54.log$

The *configuration name* is selected by setting the **other** field in the **boot parameters**. This will in turn determine the names of the files in the work directory. Ensure that there is a configuration file which correspond to the chosen configuration name.

It must be ensured that the user who "owns" the VxWorks system, has read and write access to this directory and its files. The work directory path name may be changed through startup.cmd. The UNIX and VxWorks work paths must be physically the same. In the Eurogam phase 2 setup the path names are also identical. The UNIX work path tells **cpp** where to find the Autofill work directory when it is invoked as a remote shell during configuration generation.

Here is an example of a work directory:

```
total 496
-rwxrwxrwx   1 jh        eurogam      107 May 12  1994 afdate
-rwxrwxr-x   1 jh        eurogam     1489 Dec  9 12:42 afmon
-rwxrwxr-x   1 jh        eurogam     2341 Nov 28 13:58 afview
-rw-rw-r--   1 jh        eurogam    10762 Dec 14 18:39 egam2.src
-rw-rw-rw-   1 VxWorks   eurogam    11886 Jan  5 18:24 egam2C.bin
-rw-rw-rw-   1 VxWorks   eurogam    11886 Nov 29 13:35 egam2O.bin
-rw-rw-rw-   1 VxWorks   eurogam    11886 Jan  5 17:44 egam2P.bin
-rwxrwxr-x   1 jh        eurogam     1006 Dec 21 10:59 lcheck
-rwxr-xr-x   1 jh        eurogam      511 Nov 30 16:07 netcontrol
-rw-r--r--   1 VxWorks   eurogam      125 Nov 28 13:13 passwd
-rw-rw-rw-   1 VxWorks   eurogam     2082 Dec 19 18:03 pos01.log
-rw-rw-rw-   1 VxWorks   eurogam     2084 Jan  5 18:04 pos02.log
-rw-rw-rw-   1 VxWorks   eurogam     2075 Jan  5 18:06 pos03.log
-rw-rw-rw-   1 VxWorks   eurogam     2090 Nov 26 18:05 pos04.log
-rw-rw-rw-   1 VxWorks   eurogam     2078 Jan  5 18:09 pos05.log
-rw-rw-rw-   1 VxWorks   eurogam     2079 Jan  5 18:04 pos06.log
-rw-rw-rw-   1 VxWorks   eurogam     2082 Jan  5 18:07 pos07.log
-rw-rw-rw-   1 VxWorks   eurogam     2073 Jan  5 18:08 pos08.log
-rw-rw-rw-   1 VxWorks   eurogam     2057 Jan  5 18:13 pos09.log
-rw-rw-rw-   1 VxWorks   eurogam     2083 Jan  5 18:07 pos10.log
-rw-rw-rw-   1 VxWorks   eurogam     2082 Jan  5 18:05 pos11.log
-rw-rw-rw-   1 VxWorks   eurogam     2084 Jan  5 18:07 pos12.log
-rw-rw-rw-   1 VxWorks   eurogam     2058 Jan  5 18:04 pos13.log
-rw-rw-rw-   1 VxWorks   eurogam     2086 Jan  5 18:04 pos14.log
-rw-rw-rw-   1 VxWorks   eurogam     2084 Jan  5 18:04 pos15.log
-rw-rw-rw-   1 VxWorks   eurogam     2082 Jan  5 18:06 pos16.log
-rw-rw-rw-   1 VxWorks   eurogam     2082 Dec 22 18:18 pos17.log
-rw-rw-rw-   1 VxWorks   eurogam     2084 Dec 19 07:38 pos18.log
-rw-rw-rw-   1 VxWorks   eurogam     2084 Jan  5 18:07 pos19.log
-rw-rw-rw-   1 VxWorks   eurogam     2084 Jan  5 18:05 pos20.log
-rw-rw-rw-   1 VxWorks   eurogam     2084 Jan  5 18:06 pos21.log
-rw-rw-rw-   1 VxWorks   eurogam     2092 Jan  5 18:06 pos22.log
-rw-rw-rw-   1 VxWorks   eurogam     2072 Jan  5 18:07 pos23.log
-rw-rw-rw-   1 VxWorks   eurogam     2084 Jan  5 18:07 pos24.log
-rw-rw-rw-   1 VxWorks   eurogam     2018 Jan  5 18:09 pos25.log
-rw-rw-rw-   1 VxWorks   eurogam     2082 Jan  5 18:04 pos26.log
-rw-rw-rw-   1 VxWorks   eurogam     2084 Nov 26 18:05 pos27.log
-rw-rw-rw-   1 VxWorks   eurogam     2098 Jan  5 18:04 pos28.log
-rw-rw-rw-   1 VxWorks   eurogam     2079 Jan  5 18:05 pos29.log
-rw-rw-rw-   1 VxWorks   eurogam     2090 Jan  5 18:05 pos30.log
-rw-rw-rw-   1 VxWorks   eurogam     2082 Jan  5 18:20 pos31.log
-rw-rw-rw-   1 VxWorks   eurogam     2098 Dec 16 07:45 pos32.log
-rw-rw-rw-   1 VxWorks   eurogam     2101 Jan  5 18:22 pos33.log
-rw-rw-rw-   1 VxWorks   eurogam     2134 Dec 14 08:23 pos34.log
-rw-rw-rw-   1 VxWorks   eurogam     2055 Jan  5 18:22 pos35.log
```

```
-rw-rw-rw-   1 VxWorks  eurogam     2082 Dec 21 07:11 pos36.log
-rw-rw-rw-   1 VxWorks  eurogam     2084 Jan  5 18:20 pos37.log
-rw-rw-rw-   1 VxWorks  eurogam     2032 Jan  5 07:28 pos38.log
-rw-rw-rw-   1 VxWorks  eurogam     2073 Jan  5 18:21 pos39.log
-rw-rw-rw-   1 VxWorks  eurogam     2082 Jan  5 18:25 pos40.log
-rw-rw-rw-   1 VxWorks  eurogam     2079 Jan  5 18:18 pos41.log
-rw-rw-rw-   1 VxWorks  eurogam     2055 Jan  5 18:21 pos42.log
-rw-rw-rw-   1 VxWorks  eurogam     2082 Jan  5 18:21 pos43.log
-rw-rw-rw-   1 VxWorks  eurogam     2091 Jan  5 18:20 pos44.log
-rw-rw-rw-   1 VxWorks  eurogam     2076 Jan  5 18:21 pos45.log
-rw-rw-rw-   1 VxWorks  eurogam     2084 Jan  5 18:20 pos46.log
-rw-rw-rw-   1 VxWorks  eurogam     2082 Jan  5 18:19 pos47.log
-rw-rw-rw-   1 VxWorks  eurogam     2083 Jan  5 18:23 pos48.log
-rw-rw-rw-   1 VxWorks  eurogam     2111 Jan  5 07:20 pos49.log
-rw-rw-rw-   1 VxWorks  eurogam     2077 Jan  5 18:21 pos50.log
-rw-rw-rw-   1 VxWorks  eurogam     2086 Jan  5 18:20 pos51.log
-rw-rw-rw-   1 VxWorks  eurogam     2064 Jan  5 18:23 pos52.log
-rw-rw-rw-   1 VxWorks  eurogam     2066 Jan  5 18:19 pos53.log
-rw-rw-rw-   1 VxWorks  eurogam     2079 Jan  5 18:23 pos54.log
```

As one can see from the filenames, the name of this configuration is **egam2**. The pos??.log files hold information about the latest fill times, they support the FT command in the Autofill user program.

2.2.3 HELP DIRECTORY

This directory contains one text file for each of the user commands. The other help topics also have their corresponding text files.

The default name of this directory is */eg/autoFill/help*, but it might be set to another path through **afSetHelpPath** in startup.cmd. This option will be useful if you want a choice of languages. There could e.g be a directory called */eg/autoFill/help/french* and another called */eg/autoFill/help/english*.

The following files must be present:

```
admin delay help preset threshold
avoid dgen insert quit to
cancel display iousg remove unavoid
clear fill normal show working
conn ftimes positions syntax
```

2.2.4 OBJECT FILE DIRECTORY

This is where all the object code for the Autofill resides. Presumably all Eurogam object code will be stored here. The default is */eg/obj* but may be changed if you for instance want to run an alternative software version without having to move the files to the standard directory. This is achieved by changing **objpath** in startup.cmd.

The following object files must be present in the chosen directory:

```
afExec.o afRegs.o afUser.o afXvme560.o
afLogin.o afShare.o afXvme240.o vxShare.o
afMain.o afXvme260.o
```

2.3 CONFIGURING THE AUTOFILL

The configuration and work files for the EUROGAM AUTOFILL are stored in the directory

/eg/autoFill/work

The **name** of the configuration is selected by setting the field **other** in the VxWorks boot parameters.

EXAMPLE: The chosen name is **egam2**. This will select the names of the system files as follows:

egam2.src text file containing a description of the configuration

egam2O.bin binary file containing a configuration and state database as described in egam2.src when Dgen is run.

egam2C.bin continuous backup of the database

egam2P.bin periodic backup of the database

By changing the **other** field you can easily switch between configurations. This will not be done normally, but might be useful if you want to run a test setup.

The program **afDgen** "compiles" the *.src file and leaves the resulting output in *O.bin. It can be invoked from the USER program (command DGEN) or from the VxWorks shell.

When the Autofill System starts after a reboot it will look for the database in the *egam2C.bin* first, if that fails in *egam2P.bin* and finally in *egam2O.bin*. If none of these exist the last resort will be to generate a database from *egam2.src* (calling afDgen). If that fails the system will go down, and after a short wait display the login window, indicate "DOWN" and start bleeping.

When you run the Autofill first time after installing it, no *.bin files will be present so the **afDgen** will be run automatically.

2.3.1 Editing the configuration

The source file of the configuration is a text file which MAY contain C-language preprocessor definitions. These do not have to be included but using them will make maintenance a lot easier as you can make use of the facilities of the C-language pre-processor.

The format of the configuration source is as follows:

```
System name (up to 32 characters)
Generation date (up to 24 characters)
Generation time (up to 12 characters)
Refill delay
Number of dewars
```

```
For each dewar:

    name
    sensor address and channel number

Number of detectors

For each detector:

    information string
    physical position
    presence flag
    sensor reference voltage (millivolts)
    maximum fill time (seconds)
    valve closure delay (seconds)
    preset fill times ( format HH MM ... 99 99 )
      this may be none (ie. 99 99 alone on the line)
      if 88 88 is given the fill times in the current configuration are
      preserved!
    address and channel number of liquid sensor input
    address and channel number of valve control output
    address and channel number of bias shutdown input
    manifold connection number (0-11)
    exhaust valve number (0-3)

Number of manifolds

For each manifold:

    address and channel number of inlet valve control
    address and channel number of outlet valve control
    address and channel number of liquid sensor input
    sensor reference voltage (millivolts)
    maximum purge time (seconds)
    valve closure delay (seconds)
    exhaust valve number (0-3)

Number of exhaust valves:

For each exhaust valve

    address and channel number of valve control
```

2.4 Installation checklist

1. Configure the VME cards as prescribed in chapter 2.
2. Log on to the UNIX file server as *root, eg OR VxWorks.*
3. Load the system from the delivery tar-file. The target directories listed below need not reside at the top of your directory structure. At Liverpool, for instance */eg* and */eg* are just links onto */usr/local.*
4. Check that the following directories and files are present with correct permissions:

```
drwxrwsr-x  /eg/boot
drwxrwsr-x  /eg/boot/egaf
-rw-rw-r--  /eg/boot/egaf/startup.cmd

drwxrwsr-x  /eg/autoFill

drwxr-sr-x  /eg/autoFill/help
-rw-r--r--  /eg/autoFill/help/admin
-rw-r--r--  /eg/autoFill/help/avoid
-rw-r--r--  /eg/autoFill/help/cancel
-rw-r--r--  /eg/autoFill/help/clear
-rw-r--r--  /eg/autoFill/help/conn
-rw-r--r--  /eg/autoFill/help/delay
-rw-r--r--  /eg/autoFill/help/dgen
-rw-r--r--  /eg/autoFill/help/display
-rw-r--r--  /eg/autoFill/help/fill
-rw-r--r--  /eg/autoFill/help/ftimes
-rw-r--r--  /eg/autoFill/help/help
-rw-r--r--  /eg/autoFill/help/insert
-rw-r--r--  /eg/autoFill/help/iousg
-rw-r--r--  /eg/autoFill/help/normal
-rw-r--r--  /eg/autoFill/help/positions
-rw-r--r--  /eg/autoFill/help/preset
-rw-r--r--  /eg/autoFill/help/quit
-rw-r--r--  /eg/autoFill/help/remove
-rw-r--r--  /eg/autoFill/help/show
-rw-r--r--  /eg/autoFill/help/syntax
-rw-r--r--  /eg/autoFill/help/threshold
-rw-r--r--  /eg/autoFill/help/to
-rw-r--r--  /eg/autoFill/help/unavoid
-rw-r--r--  /eg/autoFill/help/working

drwxrwsr-x  /eg/autoFill/work
-rwxr--r-x  /eg/autoFill/work/egam2.src
-rwxrwxr--  /eg/autoFill/work/lcheck
-rwxrwxr--  /eg/autoFill/work/afmon
-rwxrwxr--  /eg/autoFill/work/afview
```

```
-rwxrwxr--  /eg/autoFill/work/netcontrol
-rw-rw-r--  /eg/autoFill/work/passwd

drwxrwsr-x  /eg/obj
-rwxr-xr-x  /eg/obj/afExec.o
-rwxr-xr-x  /eg/obj/afLogin.o
-rwxr-xr-x  /eg/obj/afMain.o
-rwxr-xr-x  /eg/obj/afRegs.o
-rwxr-xr-x  /eg/obj/afShare.o
-rwxr-xr-x  /eg/obj/afUser.o
-rwxr-xr-x  /eg/obj/afXvme240.o
-rwxr-xr-x  /eg/obj/afXvme260.o
-rwxr-xr-x  /eg/obj/afXvme560.o
-rwxr-xr-x  /eg/obj/vxShare.o
```

5. Change the ownership of the above files to the user which will "own" your VxWorks system in the Autofill crate (default *VxWorks*.
6. Change the path name **/eg/boot/egaf** to correspond to the name of your crate (only if you use a non-standard setup).
7. Insert the links to vxWorks in this directory.
8. Edit the *passwd* file to suit your own requirement.
9. Edit the *egam2.src* file to reflect your own hardware and your connection between the array and the Autofill crate.
10. Edit startup.cmd in the work directory. You will need to change the *server* and *hostAdd entries.*

    You MUST set the following **site dependent parameters**:

    Objpath, rloginhost, rshell host(set to server), unixworkpath, helppath, cpppath.

    If you adhere to the standard file organisation you should not need to edit any of these except "rloginhost". You MUST use the function calls provided like it is done in the **startup.cmd** delivered on the tar-file.
11. Edit the boot parameters.

```
Example of a boot-line (Strasbourg):

boot device          : ln
processor number     : 0
host name            : egc5 *
file name            : /eg/boot/egaf/vxWorks ?
inet on ethernet (e) : 193.48.87.7 *
host inet (h)        : 193.48.87.25 *
user (u)             : VxWorks        ?
flags (f)            : 0x6
target name (tn)     : egaf ?
```

```
startup script (s)   : /eg/boot/egaf/startup.cmd ?
other (o)            : egam2 *
```

The fields marked '*' will have to be changed, the **egaf** entries all refer to the cpu name. The entries marked '?' will need to change if you use a non-standard setup. The name of the configuration is **egam2**, the server "egc5" and the CPU name "egaf".

12. Make sure the user (u) field in the boot parameters and the ownership of the Autofill related files correspond.
13. Continue with the booting

2.5 Booting first time

If your startup.cmd executes its calls correctly, you will get the message

EUROGAM AUTOFILL STARTING

on the screen, and as the configuration is generated messages describing its progress are displayed. This might take a few seconds. Then the messages from the Autofill startup sequence will appear on the screen:

```
LOADING tasks ..     OK
STARTING tasks ..    OK
STARTING selftest .. OK
```

This message will stay put for a few seconds, then login will be initiated. The system is now ready for normal operation, and you can log on. For further information, refer to the Autofill users manual.

3 SYSTEM MAINTENANCE

This section deals with the day to day running of the Autofill system, like maintaining the work directory. The maintenance can be don either from the VxWorks shell or from a SUN workstation. The latter is easier, and a few hints are given below. All the examples assume that the configuration name is "egam2".

3.1 WORK DIRECTORY

This is where you find the files that are used by the Autofill in its operational phase. It contains backup, log files, configuration source and a few supplement programs to facilitate remote viewing of the system state.

The database backup files are

- egam2O.bin - the original database as generated when Dgen was last run
- egam2P.bin - the periodic backup
- egam2C.bin - the "on demand" continuous backup

The supplement programs are:

- lcheck - views statistics of previous fills, by default the last fill.
- afview - view system, either by detector, manifold or parameter
- afmon - continuously display the main systems status
- afdate - set date/time of Autofill identical to server
- netcontrol - switch Autofill to/from standalone mode, this programs may be used by the server to disconnect/reconnect the Autofill safely when it (the server) is taken down/rebooted.

3.1.1 MODIFYING THE CONFIGURATION

You might sometimes want the change the configuration of the Autofill by editing the file ¡configname¿.src. To do this you have to perform the following:

- Edit the source file in the work directory.
- Log on to the Autofill console as a privileged user
- Do the ADMIN command, followed by DGEN
- Confirm, when asked if you want to reboot

- If this reboot fails, boot manually
- After reboot, log on and UNAVOID the detectors you want to use

The old database gets stored in old.bin and can be retrieved if you made a mistake.

You can also do this remotely from a work station as follows:

```
edit the source file
ersclaim egaf
erswrite control -a afHalt
mv egam2C.bin old.bin
rm egam2?.bin
erswrite control -a reboot
```

RESTORING AN OLD DATABASE: If you have saved an old database in say "old.bin" and want to restore it do the following (on the SUN):

```
ersclaim egaf
erswrite control -a afHalt
cp old.bin egam2C.bin
erswrite control -a reboot
ersfree

OR from the VxWorks shell:

afHalt
copy "old.bin","egam2C.bin"
reboot 6
```

3.2 REGISTERS

Some Eurogam registers have been implemented, mostly for the purpose of allowing remote viewing of the system. These registers are currently used by the *afview* and *afmon* commands.

The register "control" was added for executing single-argument commands in the crate.

Example: erswrite control -a reboot (careful!!)

A useful debug commands is: **erswrite control -a vxinfo**

This will execute the VxWorks commands **i**, **checkStack** and **memShow** and dump the output in the file "vxinfo.d" in the work directory.

To obtain information about the current state of the database, type:

```
erswrite control -a dbSave
```

This stores a description of the database in the file "afdb.d" in the work directory.

3.3 SETTING THE CLOCK

The clock can be set from the VxWorks shell through the command **setDate**. This command allows you to type in the data and the time. A simpler method is to use **rdate**. This will pick up the time of the SUN server and set the Autofill time accordingly.

rdate is always executed when the system is booted. The VxWorks shell command **printDate** displays current time and date..

NB! Remember the summertime. All you have to do is to execute an *rdate* when you are positive that the servers time has changed over.

From a SUN: *afdate*

From the VxWorks shell: *rdate*

Or: *Reboot the crate* !

3.4 DEBUG VARIABLES

Four debug variables have been included to enable and disable some functions of the system. They are defined as follows in the file "debug.c" in the /eg/autoFill/src/share directory.

```
PUBLIC int afDebug= 0;
/* bit  0 - force load user program for each login */
/* bit  1 - show hex value of detector/manifold status in show command */

PUBLIC int enableBs= 1;   /* Enable bias shutdown detection */
PUBLIC int enableKey= 1;   /* Enable manual fill key         */
PUBLIC int enableLog= 1;   /* Enable message logging         */
```

You can change those as you go by logging in to the VxWorks shell, or set them to different values in "startup.cmd". Note that disabling message logging will improve the response time for user commands. The default values are as shown in the example above.

You can enable and disable the message logging from the SUN as follows:

```
erswrite control -a msgOn
erswrite control -a msgOff
```

4 CODE STRUCTURE

The Autofill software takes advantage of the VxWorks multi-tasking facility by dividing the software into a number of tasks according to which function of the system they perform. This provides for modular software and hence easier maintenance.

The core of the Autofill system consists of four tasks which communicate via a shared data module hereafter referred to as the *database*. The database contains configuration and state information for each detector and each manifold. Three of these tasks are drivers for the three types of IO-cards used by the system, while the EXEC task controls the fill cycle.

The four **core tasks** are:

- XVME240: driver for bias shutdown input, it reads the TTL-inputs on the XVME240 card, and writes the values into the database. One input is set aside to sense the state of the *Manual Fill* key on the manifold control units. One channel is configured as output and is used to inhibit the filling of the main 300 liter liquid nitrogen dewar during and after a fill.
- XVME260: driver for the relay card which controls the valves. It reads the required valve state from the database (these are provided by EXEC), and ensures that the valves are operated accordingly.
- XVME560: driver for liquid sensor input, it reads the ADC values from the XVME560 card, and stores the values in the database for subsequent use by EXEC.
- EXEC: takes care of fill operations etc. This is the "heart" of the Autofill system. It communicates with the drivers via the database. This task constantly checks the timing information in the database to determine whether detectors need filling. When a fill is due EXEC initiates it by telling the XVME260 task to open the relevant valves. Then it will continuously check ADC-values from the XVME560-task to determine whether the detector is full or not. The EXEC task always checks for *fill timeouts* (UNDERFULL), and *liquid exhaust* when valves should be closed (OVERFULL). EXEC also controls the input and output valves of the manifolds. The database is backed up to disk to enable reconstruction of the system after power failure.

Operator access to the Autofill is provided by

- USER: this is a program to provide the operator with means to intervene in the fill cycle, set new fill times and get a detailed view of the state of the Autofill system. This program also uses the database, some of the parameters can be changed by a privileged user. For details see the Autofill Users Manual.

- LOGIN: login daemons serving the four serial ports. These will call the USER program after a correct login identifier and password have been given. Access to the VxWorks shell and a remote SUN workstation is also provided for.

MAIN starts the system and also supervises the other tasks, constantly checking for fatal error conditions.

Intertask communication is done through the following mechanisms:

- SEMAPHORES: used for synchronisation. The scheduling of the *core tasks* is controlled by a synchronisation semaphore with a timeout in order to avoid deadlock.
- DATABASE: large buffer containing the setup and the state of the Autofill, read and written by the four *core* tasks and USER. Protected by the synchronisation semaphores.
- SHARED VARIABLES: In the code these are hidden by the use of macro definitions. They are defined in header files by the keyword SEXTERN. One of the shared variables **Afstate** contains the overall system state. It is always displayed on all four terminals, regardless of whether the user is logged on or not. See Autofill Users Manual.

4.1 STARTUP

After VxWorks is booted it runs a script *startup.cmd* which is in fact the application specific part of the boot sequence. Except for the choice of Autofill configuration name, all parameters relevant to the Autofill are set in this script (see section 1).

- load common Eurogam startup code
- load shared code (still application independent)
- initialises crate specific parameters.
- setSever sets the name of the current server
- setRloginHost determines which remote SUN you will connect to if you rlogin directly from the login prompt.
- setRshellHost determines which SUN should execute the remote commands like RDATE and DGEN. This should be set to server.
- setObjPath decides where the the object code should be loaded from
- rdate syncs the crate clock with the server

- afInitCrate sets a map of the XVME cards and initialises them
- afSetUnixWorkPath determines where the files created by afDgen should be stored. This as seen from the work station used by afDgen for it's cpp call. This path must physically correspond to the workpath, even though the path **names** may vary depending on how the file systems are mounted.
- afSetHelpPath tells the USER program where to look for help files
- afSetCppPath determines which **cpp** to use (as seen from workstation)
- change to working directory and start Autofill

4.2 TIMING

The Autofill uses the on-board clock of the MVME147 to generate time references. For historical reasons (the system was migrated from OS/9) it uses a (quasi) Julian time, i.e counting seconds from an absolute reference.

4.3 DATABASE

The database consists of three sections:

- general information (name, creation date, etc.)
- state information (for each detector and manifold)
- configuration information (for each detector and manifold)

The **state** of a detector is described by

- Status word, current (described below)
- Status word, at completion of last fill
- Most recent ADC-value
- delay time, if a fill has been delayed the total value of the delay is recorded (maximum 1 hour).
- timing information on latest fill
- time of next fill

In the **state** of a **manifold** is described by

- Status word (described below)
- Most recent ADC-value
- Number of open output valves on the manifold
- Number of overfull detectors on the manifold
- timing information on latest purge

The **configuration** of a **detector** is described by

- information string
- physical position
- sensor input connection (XVME560 card address, channel number)
- bias shutdown input connection (XVME240 card address, channel number)
- valve output connection (XVME260 card address, channel number)
- reference voltage
- maximum allowable fill time

- valve closure delay time
- table of pre-programmed fill times
- manifold number (0-11)
- exhaust valve number (0-3)

The **configuration** of a **manifold** is described by

- inlet valve connection (XVME260 card address, channel number)
- outlet valve connection (XVME260 card address, channel number)
- sensor input connection (XVME560 card address, channel number)
- reference voltage
- maximum allowable purge time
- valve closure delay time
- exhaust valve number (0-3)

The status word of a detector/manifold has these bits:

- OVERFULL, set if liquid has been detected when not expected to. This can i.e result from a valve being stuck in open position.
- UNDERFULL, set if no liquid has been detected within a set period after a fill has started. Most likely to be caused by ice blockage.
- FILL_ABTD, set if any detector on the manifold or the manifold itself is overfull. This leads to the manifold inlet being closed, so any ongoing fills on the affected manifold will be aborted. This flag is also set when a fill has been aborted by a power failure.
- FILLING, set when a detector is filling.
- DELAYED, set in the period between initially scheduled fill time and a new fill time resulting from a DELAY command from the user.
- PURGING, set when a manifold is being purged of air
- AVOIDED, indicates detector AVOIDED, i.e no fill actions or alarms take place.
- SHUTDOWN, indicates bias shutdown from the GE detector. This is caused by the detector warming up. The AVOIDED flag is set automatically.
- LIQUID, set when liquid nitrogen has been sensed by XVME560 card

- GAS, set when gas has been sensed by XVME560 card
- MISSED_FILL, set when a fill has been missed due to power failure
- TOG_AVOID, set by USER program to tell EXEC to toggle the AVOIDED flag.
- MANF_ERR, set if an error has occurred during purging of the manifold.
- REQ_CANCEL, set by USER program to request cancelling of a fill.
- ALARM, inclusive or of MISSED_FILL, OVERFULL, UNDERFULL and MANF_ERR flags.
- VALVE_IN, input valve state
- VALVE_OUT, output valve state (manifold only)
- DIRTY, flag telling that the state has changed and backup is required.

4.4 EMERGENCY BACKUP

The backup of the database to disk is done using two independent methods:

- backup on demand: whenever the state of the database changes, the relevant part is written to a file. It is therefore ensured that there at any moment exists an updated image of the database on disk. There is a delay of between the change and the actual update. This is to allow time to accumulate changes and thus cut down on the number of disk accesses.
- periodic backup: the database is written in full to a disk file at regular intervals. The current release uses the same disk for both backups.

4.5 MAIN TASK

As explained under STARTUP this task is spawned from the startup script. It initialises the shared variables(defined as SEXTERN in the code). The VxWorks shell is deleted to avoid it interfering with the console.

The next step is to load the code of the other tasks and start them with the desired setting of priority, options and stacksize. This includes the core tasks and the login daemons.

The task names of the Autofill application all start in "af" with corresponding entry points "_afxxxxx" in order that they be distinguishable from other tasks running on the CPU. For instance, the EXEC task will be known to VxWorks as "afExec", the XVME560 as "afXvme560" etc.

After spawning the other Autofill tasks **main** blocks until has received signals from the *core tasks* telling it that the initialisation phase has succeeded. **Main** then signals the login daemons that everything is ready and the system now enters its operational phase. From this point onwards *main* only checks that the other tasks are "alive", it does not interfere with them. If a task is found to be suspended the *core tasks* are terminated, all VME-boards are reset. This calamity could either be caused by an exception like a bus error, or a bug in the software. Any open valves will be shut and the red LEDs on the XVME-cards will come on. The system state will be DOWN.

4.6 EXEC TASK

4.6.1 INITIALISATION

This is the "control centre" the filling operations. The first operation is to load the database from disk. This will normally be a retrieval of the latest state before the system was rebooted. For first time boot, see section 1. If loading the database succeeds, the three driver tasks are signalled that the database is valid.

The state of the database is checked after loading to see whether there are missed fills or any fills interrupted by a system reset. In both cases error flags will be set and the alarm turned on. Further an initial backup is made on *both* the backup files. The initial phase is now completed and a signal is sent to **main** (to indicate "system operative").

4.6.2 OPERATIONAL PHASE

The task must first wait for its execution timeslot. Then each detector is repeatedly checked as follows:

- check for **avoid** or change in avoid setting
- if not avoided:
  - if time for fill: open valve(s)
  - check for cancellation request
  - check for exhaust liquid; if seen then close valve(s) **or** set OVERFULL flag depending on whether detector is filling or not. If no liquid is seen within a timeout, the detector state is set UNDERFULL
  - check the corresponding manifold; operate valves and set error flags if necessary

Check if detector needs backup.

During an execution slots all the detectors are looped through to check for fills and error conditions. At the end of the slot, any necessary backup will be done.

4.6.3 FILL CYCLE

The fill cycle for a detector starts when EXEC determines that current time has reached the programmed fill time. Initially all alarm conditions for that detector are cleared and the fill is initiated by setting the status flag PURGING. If there are no other detectors filling on the manifold, the PURGING flag of the manifold is also set together with the flags instructing the XVME260 task to open the manifold input and output valves. If the manifold is already purging, no action is taken on it. If another detector on the same manifold is filling, and the manifold is already purged the fill will start straight away.

When liquid is seen at the manifold output, the output valve closes and simultaneously all the detectors which have requested a fill, ie have their PURGING flag set, get their inlet valves opened and enter the FILLING state.

The fill may be stopped by the commands "CANCEL" or "AVOID" in the user program, but would normally go on until liquid is seen at the LN2 exhaust point. The manifold inlet valve is kept open as long as there are any detectors still filling. A fill could also be terminated if liquid is not seen within a set period. In this case the UNDEFULL alarm condition is set.

Another alarm condition is OVERFULL. This happens if the sensor continues to see liquid after the inlet valves ought to have closed (save a delay to account for LN2 still in the pipes etc.). This will also cause the manifold inlet valve to be closed, thus aborting the fill of detectors on the affected manifold. The manifolds "detector overfull" count will be increased. The FILL_ABTD flag is then set for the detectors whose fills have been aborted. This is an error condition, and the detectors concerned must be filled again.

The UNDERFULL and OVERFULL errors also apply to the manifold purge. If purging of a manifold times out, all the detectors concerned get their MF_ERR flag set. This also happens if the manifold is OVERFULL. The manifold overfull error causes a fill abortion on the whole manifold. All ongoing fills (one that manifold) are stopped and the FILL_ABTD flag set. When a mainfold undefull occurs, any filling detectors will get their UNDERFULL flag set.

4.7 XVME560 TASK

The purpose of this task is to read the liquid sensor voltages through an ADC, and store the value(s) in the database. Thus it is available for the EXEC task for it to determine whether or not liquid is seen.

During initialisation the task first waits for a signal from EXEC that the database is loaded from disk. Then it makes a list of all the **card addresses** found in the configuration, and associates a list of detectors with each of these cards. After this the cards are tested (for presence in crate) and initialised. There is also a simple test of the validity of the configuration(e.g. checking that no two detectors are configured to use the same ADC-channel). The result of the initial tests are logged. If any of the tests fails the task will terminate and the Autofill system will go down.

In the operational phase each card is handled successively, the ADC-value for each detector on the card are read and stored in the database. All cards are scanned before the task releases its execution slot.

4.8 XVME240 TASK

This structure of this task is exactly as for XVME560, but in the main loop this task polls the digital input for bias shutdown and writes the value to the database. This task also reads the key-switches on the front of the manifold control units so the computer can take action to inhibit interference from the console terminal. One of the channels on the XVME240s is configured as an output, and drives the "dewar fill inhibit" signal. In the current setup fill of the main dewar is inhibited *during fills* and 1 hour after.

4.9 XVME260 TASK

This task is structured as the two other drivers. It *reads* values from the database and and sets the outputs on the XVME260 cards accordingly. The state of the relay driver outputs on this card reflects the state of the VALVE_IN and VALVE_OUT flags in the database.

4.10 LOGIN DAEMONS

4.10.1 INITIALISATION

Unlike the other tasks, the four login daemons are not spawned directly by **main**. Main starts a single **login** task which opens and initialises those tty-devices which are not used by VxWorks. This task now loads the password file and spawns the four *login daemons* before it terminates. The login daemons are identical, re-entrant tasks.

4.10.2 OPERATION

Each daemon is initialised by setting its standard io, terminal options, port name, number etc. Then the display is initialised and execution pends reception of a signal from **main** when the initialisation of the four *core tasks* has completed. The login prompt is then displayed and normal login operations can take place. Privileged access to the Autofill, VxWorks shell login, remote server login are only allowed on the console. The login daemons constantly checks the state of the system and maintains a display of this. If a filling error or a system crash occurs, the login daemons will send continuous bleeps to the connected screens.

The main purpose of the login daemon is to provide controlled access to the **user** program (see the next paragraph). **User** is loaded from disk during daemon startup and an entry pointer is kept for subsequent sessions. Each daemon keeps its own copy of **user** as it (user) is not re-entrant.

4.11 USER PROGRAM

The **user** program is the human interface to the Autofill system. Through this an operator can manipulate valves, change fill times, avoid detectors, and observe the state of the system in various degrees of detail. **User** is not an independent task, it is called as a subroutine from the login daemons. The *privilege level* and user name are passed as parameters. The first is significant as operations which change the database can only be done by privileged users (and only when logged on to the console). A comprehensive **online help** facility is provided. Details of the commands are described in the Autofill Users Manual (EDOC098).

5 CODE MAINTENANCE

The source code is kept on the **nsa** server at Liverpool under the directory */eg/autoFill/src*. The documentation is kept in */eg/autoFill/doc*. A copy of the source code is kept at the Daresbury machine **nnsa** and at the Strasbourg machine **orchidee**.

The code is organised in a hierarchy of directories, where the code of each task is kept separate. There are also separate directories for the USER program, library code, include files and shared code. Each directory has its own *makefile* which generate the object code. For example typing *make* in the "exec" directory, will generate the object module "afExec.o". References to shared variables and VxWorks are resolved as the code is loaded by VxWorks.

NB! BEFORE RUNNING MAKE, YOU HAVE TO SOURCE THE FILE **make_env** which can be found in **/eg/autoFill/src**. This sets the necessary environment variables.

Make also has targets for print, clean, tags and "proto.h" (prototyping). The library and shared module prototype files are copied over to "include". The complete system is generated by running *make* from **/eg/autoFill/src**.

Directory organisation:

- VXSHARE, shared code, not tied to the Autofill application
- SHARE, shared code, Autofill specific
- USER, code for USER program
- MAIN, code for MAIN task
- LOGIN, code for LOGIN tasks
- EXEC, code for EXEC task
- XVME240, code for XVME240 driver task (bias shutdown sense)
- XVME260, code for XVME260 driver task (valve operation)
- XVME560, code for XVME560 driver task (liquid sensor)
- INCLUDE, common include files + makerules
- VXLIB, general purpose library for VxWorks
- AFLIB, library for Autofill application

The rules for the make command are kept in *rules.mk* and *generate.mk* in the **include/make** directory. These contain common definitions for all the makefiles. The makefiles themselves are thus kept simple, they contain little more than a list of the files which are to be *compiled* and *linked* together. The rest is accomplished by including the "rule" files.

The **exec, xvme560, xvme260, xvme240, main and login** tasks each have separate directories.

ADDITIONAL DIRECTORIES ARE:

5.1 vxshare

Does not contain any Autofill specific code, but som of the routines found here are used. The code is tied to the MVME147 / VxWorks / SUN environment and includes bus error testing, drivers for the clock chip on the MVME147, the rdate and other commands to set the time, code to set site parameters like "object path"(see startup.cmd), code for use of serial ports on MVME147, a timer pool (which uses th VxWorks system clock), and some utility routines. There is also a SUN remote shell facility (a primitive rsh), which is used by **rdate** and the Autofill configuration compiler **afDgen**. Some Eurogam registers are included here.

A copy of the proto.h file is kept in "include" as **vxShare.h**. The **vxshare** code is also used by the Rate Monitor and HighVoltage.

5.2 vxlib

Contains library routines for to support *afDgen*, vt100 display, string manipulations, user identification, the "vxinfo" command (see Registers), and keyboard input support, VxWorks shell login and some utility functions.

A copy of the proto.h file is kept in "include" as **vxLlib.h**. The **vxlib** code is also used by the Rate Monitor and HighVoltage.

5.3 share

This is shared code available to all tasks like in "vxshare" but here tied to the Autofill. It contains the function to search the crate for XVME cards, the **afDgen** configuration tool, and the code for restarting with a blank configuration. The register configuring is also found here as well as the functions to set some additional site dependent parameters. In startup.cmd these will begin with "af". The code for interfacing to the Eurogam message logger is also included here.

A copy of the proto.h file is kept in "include" as **afShare.h**.

5.4 aflib

Library of routines tied to the Autofill application. Contains a special **fopen** which sets the SYS_ALONE flag in **Afstate** if a *fopen* does not return within a set period.

A copy of the proto.h file is kept in "include" as **afLib.h**.

5.5 user

The largest of all the modules, it contains the code for all the commands and various supporting routines like input line parsing. During debugging this module can be force-loaded on each login (set afDebug=1).

5.6 include

```
gdefs.h - general purpose definitions and macros *
gtime.h - timing related definitions *
vxLlib.h - prototypes from the library vxlib *
vxShare.h - prototypes from the shared module vxshare *

afLib.h - prototypes from the library aflib
afShare.h - prototypes from the shared module share
afVxBind.h - definitions for VxWorks - Autofill interface
afdefs.h - autofill database specific definitions
afmsg.h - definition of log messages
afvme.h - definitions for the XVME cards and the driver tasks
afdrv.h - data structures fro XVME drivers
afdbase.h - database structure for core tasks and USER
afsys.h - system state definition and site dependent parameters
```

*) These files are taken from /eg/include. This is a common Eurogam include directory.

6 APPENDICES

6.1 Example of startup.cmd

```
# ====================================================
#    Startup for Eurogam Autofill
# ====================================================
< /eg/boot/startup.cmd
objpath="/eg/obj";
workpath="/eg/autoFill/work";
cd (objpath);
ld (0,0,"message.o");
ld (0,0,"register-server.o");
ld (0,0,"vxShare.o");
#
setServer(server);
setRshellHost(server);
setRloginHost("egc5");
setObjPath(objpath);
rdate();
#
ld (0,0,"afShare.o");
afInitCrate(setleds,2);
afSetUnixWorkPath(workpath);
afSetHelpPath("/eg/autoFill/help");
afSetCppPath("/usr/local/vw/bin/solaris/cpp");
#
ld (0,0,"afMain.o");
ld (0,0,"afRegs.o");
ersInstall(0,0);
#
cd (workpath);
afStart();
```

6.2 passwd format

```
username password group

Group:

0 - nonprivileged Autofill access
1 - privileged Autofill access
3 - VxWorks shell access
4 - login to a remote work-station
(determined by setRloginHost in startup.cmd)
```

6.3 DETECTOR STATUS WORD

Extract from **afdefs.h**

```
/* D - refers to use in detector status word , M to manifold */

enum {  N_OVERFULL=0, N_UNDERFULL, N_FILL_ABTD,
        N_MANF_ERR, N_MISSED_FILL,
        N_FILLING,  N_DELAYED, N_PURGING,       N_CLOSING,
        N_LIQUID,   N_GAS,     N_WARNING,
        N_AVOIDED,   N_SHUTDOWN, N_OVERRIDE,
        N_WARN_RANGE,
        N_WARN_FILLT, N_WARN_CANCEL, N_WARN_NOGAS,
        N_VALVE_IN,  N_TOG_AVOID, N_REQ_CANCEL,
        N_PRESENT,   N_DIRTY, N_DT_OVERF , N_SCHEDFILL};

#define OVERFULL        (1<<N_OVERFULL)         /* D M  */
#define UNDERFULL       (1<<N_UNDERFULL)        /* D M  */
#define FILL_ABTD       (1<<N_FILL_ABTD)        /* D    */
#define FILLING         (1<<N_FILLING)          /* D    */
#define DELAYED         (1<<N_DELAYED)          /* D    */
#define PURGING         (1<<N_PURGING)          /* D M  */
#define CLOSING         (1<<N_CLOSING)          /* D M  */
#define VALVE_OUT       (1<<N_PURGING)          /* D M  */
#define AVOIDED         (1<<N_AVOIDED)          /* D    */
#define SHUTDOWN        (1<<N_SHUTDOWN)         /* D    */

#define LIQUID          (1<<N_LIQUID)           /* D M  */
#define WARNING         (1<<N_WARNING)
#define MISSED_FILL     (1<<N_MISSED_FILL)      /* D    */
#define MANF_ERR        (1<<N_MANF_ERR)         /* D    */
#define VALVE_IN        (1<<N_VALVE_IN)         /* D M  */
#define PRESENT         (1<<N_PRESENT)          /* D    */
#define DIRTY           (1<<N_DIRTY)            /* D M  */
#define TOG_AVOID       (1<<N_TOG_AVOID)        /* D    */
#define REQ_CANCEL      (1<<N_REQ_CANCEL)       /* D    */
#define WARN_FILLT      (1<<N_WARN_FILLT)       /* D    */
#define WARN_CANCEL     (1<<N_WARN_CANCEL)      /* D    */
#define WARN_NOGAS      (1<<N_WARN_NOGAS)       /* D    */
#define WARN_RANGE      (1<<N_WARN_RANGE        /* D    */
#define GAS             (1<<N_GAS)              /* D    */
#define OVERRIDE        (1<<N_OVERRIDE)         /* D    */
#define SCHEDFILL       (1<<N_SCHEDFILL)        /* D    */
```

6.4 DATABASE DEFINITION

```
/* DETECTOR data definition */

typedef struct {

    /*  Configuration (static data) */

    char detname[DNAMLEN]; /* Information string */
    Count phypos; /* Physical position in detector array */

    SVMEadr adcptr; /* ADC card base address */
    Offset aoff; /* ADC input channel # */
    SVMEadr rva; /* Detector valve XVME260 base address */
    Offset rvoffa; /* Detector valve channel # */
    SVMEadr bias; /* Bias shutdown XVME 240 base address */
    Offset biasoff; /* Bias shutdown channel # */
    Count manfno; /* Manifold channel number (0...MAX_MANF) */
    Count exhstNo; /* Exhaust valve number (0..MAX_EXHAUST) */

    Voltage vref; /* Sensor reference voltage (mv) */
    Jtime tfm; /* Max allowable fill time (sec) */
    Jtime supt; /* Valve closure delay time (sec) */

    /*  Detector state (volatile data) */

    long s; /* Status word current (see afdbase.h) */
    long cs; /* Status word (end of last fill) */

    Voltage vadc; /* Current ADC value (mv) */

    Jtime tdld; /* Accumulated delay time (sec) */
    Jtime rtvo; /* Local time when valves last opened */
    Jtime rtnf; /* Local time of next fill */
    Jtime rtvc; /* Local time when valve closed */

    Ltime irtvo; /* 00HHMMSS versions of some of the above */
    Ltime irtnf;
    Ltime irtvc;

    Fttab rtfcs; /* Desired fill times in form HHMM00  */
    Ltime ifdat; /* Date of last fill: YYMMDD */

} Detector;

/*  MANIFOLD data definition */

typedef struct {
```

```
    SVMEadr rvb; /* Inlet valve XVME260 base address */
    Offset rvoffb; /* Inlet valve channel # */
    SVMEadr rvc; /* Purge valve XVME260 base address */
    Offset rvoffc; /* Purge valve channel # */
    SVMEadr adcptr; /* ADC card base address (purge sensor) */
    Offset aoff;     /* ADC input channel # (purge sensor) */

    Voltage vref; /* Sensor reference voltage */
    Jtime tfm; /* Max. allowable purge time (sec) */
    Jtime supt;      /* Purge closure delay time (sec) */
    Count exhstNo;      /* Exhaust valve number (0..MAX_EXHAUST)
*/

    Voltage vadc; /* Most recent ADC-value */
    long s; /* Status word (ref. afdefs.h) */
    Count nOpen; /* Number of open detector valves */
    Count nOfull; /* Number of overfull detectors on manifold
*/

    Jtime rtvo; /* Local time when valves last opened */
    Jtime rtvci; /* Local time when inlet last closed */
    Jtime rtvco; /* Local time when purge last closed */
    Ltime irtvo;        /* 00HHMMSS versions of the above */
    Ltime irtvc;
    Ltime ifdat; /* Date of last fill: YYMMDD */
} Manifold;

/* Exhaust valve definition */

typedef struct {
    SVMEadr rve; /* Exhaust valve XVME260 base address */
    Offset rvoffe;      /* Exhaust valve channel # */
    Count nOpen; /* Number of open detector valves */
    Count det[MAX_DET_EXHAUST]; /* Physical positions of detectors
*/
    Count man[MAX_MAN_EXHAUST]; /* Manifold numbers */
} Exhaust;

typedef struct {
    char name[DNAMLEN];
    SVMEadr dew_addr; /* XVME560 address */
    SVMEadr dew_offs; /* channel number */
    Voltage vadc; /* Most recent ADC-value */
    Voltage rlow,rhigh; /* Range (low .. high; assumed linear
*/
} Dewar;
```

```
/* Overall DATABASE structure */

typedef struct {

    char sysnam[SNAMLEN]; /* Name of this configuration */
    char fnam[FILNAML]; /* Source file name */
    char crdate[DATELEN]; /* Creation date */
    char crtime[TIMELEN]; /* Creation time */

    Count nf; /* Number of detectors being filled */
    short refill_delay; /* Delay of refill after power failure
*/
    SVMEadr ffptr; /* Address of card for fill flag output */

    Count nDewars;
    Dewar dewars[MAX_DEWARS];

    Count nDets;
    Detector dets[MAX_DETS];

    Count nManf;
    Manifold manfs[MAX_MANF];

    Count nExhaust;
    Exhaust exhst[MAX_EXHAUST];

} DataBase;
```

6.5 SYSTEM STATUS WORD

An extract from the file **afsys.h** defining the bits in the system status word:

```
enum {  S_RUN=0, S_TESTING, S_ALONE,   S_BIAS_SHD, S_WARNING, S_FILL_ERR,
        S_FILL,  S_HALTED,  S_CRASHED, S_MANUAL,   NSSB };

#define SYS_RUN         (1<<S_RUN)
#define SYS_TESTING     (1<<S_TESTING)
#define SYS_ALONE       (1<<S_ALONE)
#define SYS_BIAS        (1<<S_BIAS_SHD)
#define SYS_FILLING     (1<<S_FILL)
#define SYS_WARNING     (1<<S_WARNING)
#define SYS_FILLERR     (1<<S_FILL_ERR)
#define SYS_HALTED      (1<<S_HALTED)
#define SYS_CRASHED     (1<<S_CRASHED)
#define SYS_MANUAL      (1<<S_MANUAL)
```

6.6 CONFIGURATION SOURCE

```
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*
*    A u t o f i l l     c o n f i g u r a t i o n    f i l e
*
*    Author: Jomar Honsi       Eurogam phase 2 SWREV 2.4
*
*    Date:   28/11/94
*
*    Filename:  egam2.src
*
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */

#define PRESENT 1
#define NOT_PRESENT 0
#define NO_REFILL -1
#define NO_DEWAR -1
#define NO_EXHAUST -1

/* Manifold numbers (must be 0-11) */

#define MANF_A  0
#define MANF_B  1
#define MANF_C  2
#define MANF_D  3
#define MANF_E  4
#define MANF_F  5
#define MANF_G  6
#define MANF_H  7
#define MANF_I  8
#define MANF_J  9
#define MANF_K  10
#define MANF_L  11

/* Exhaust valve number (0 ... )   */

#define EXHAUST_1       0
#define EXHAUST_2       1
#define EXHAUST_3       2
#define EXHAUST_4       3

/* Addresses of XVME560 analogue input cards */

#define ACARD1 4000
#define ACARD2 4400
#define ACARD3 4800
```

```
/* Addresses of XVME240 TTL-level digital IO cards */

#define DCARD1 2000
#define DCARD2 2400

/* Addresses of XVME260 relay output cards */

#define RCARD1 0400
#define RCARD2 0800
#define RCARD3 1000

/* Dectector settings */

#define REFV       -60 /* Reference voltage (millivolts) */
#define MAXFT      1500.0 /* 25 minutes */
#define MAXFT_L    1800.0 /* 30 minutes */
#define SUPT       75.0 /* Closure delay time (seconds) */
#define FILL_1 07 30 18 00 99 99
#define FILL_2 FILL_1
#define FILL_3 FILL_1
#define OLDFILLTIMES 88 88
#define NOFILLTIMES  99 99

/*   Data starting here */

EUROGAM PHASE 2
__DATE__
__TIME__

30 /* Refill delay (minutes) */

2 /* Number of dewars */
Dewar(L)
ACARD1 6
Dewar(R)
ACARD3 30

72 /* Number of detector channels */
/*============
**  MANIFOLD A   **
**===========*/
Detector 14
14
PRESENT
REFV
MAXFT
```

```
SUPT
FILL_3
ACARD1 0
RCARD1 0
DCARD1 0
MANF_A
EXHAUST_1

Detector 15
15
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 1
RCARD1 1
DCARD1 1
MANF_A
EXHAUST_1

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 6
RCARD1 2
DCARD1 2
MANF_A
NO_EXHAUST

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 3
RCARD1 3
DCARD1 3
MANF_A
NO_EXHAUST
```

```
Detector 51
51
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 4
RCARD1 4
DCARD1 4
MANF_A
EXHAUST_1

Detector 53
53
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 5
RCARD1 5
DCARD1 5
MANF_A
EXHAUST_1

/*=============
**  MANIFOLD B   **
**============*/
Detector  5
5
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 16
RCARD1 16
DCARD1 8
MANF_B
EXHAUST_1

Detector 13
13
PRESENT
REFV
MAXFT
```

```
SUPT
FILL_3
ACARD1 17
RCARD1 17
DCARD1 9
MANF_B
EXHAUST_1

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 18
RCARD1 18
DCARD1 10
MANF_B
NO_EXHAUST

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 19
RCARD1 19
DCARD1 11
MANF_B
NO_EXHAUST

Detector 47
47
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 20
RCARD1 20
DCARD1 12
MANF_B
EXHAUST_1
```

```
Detector 49
49
PRESENT
REFV
MAXFT_L
SUPT
FILL_3
ACARD1 21
RCARD1 21
DCARD1 13
MANF_B
EXHAUST_1

/*=============
**  MANIFOLD C   **
*************/
Detector  4
4
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 8
RCARD1 8
DCARD1 16
MANF_C
EXHAUST_1

Detector 11
11
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 9
RCARD1 9
DCARD1 17
MANF_C
EXHAUST_1

Detector 12
12
PRESENT
REFV
MAXFT
```

```
SUPT
FILL_3
ACARD1 10
RCARD1 10
DCARD1 18
MANF_C
EXHAUST_1

SPARE
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 11
RCARD1 11
DCARD1 19
MANF_C
EXHAUST_1

Detector 43
43
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD1 12
RCARD1 12
DCARD1 20
MANF_C
EXHAUST_1

Detector 45
45
PRESENT
REFV
MAXFT_L
SUPT
FILL_3
ACARD1 13
RCARD1 13
DCARD1 21
MANF_C
EXHAUST_1
```

```
/*=============
**  MANIFOLD D   **
*************/
Detector 54
54
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD1 24
RCARD1 24
DCARD1 24
MANF_D
EXHAUST_2

Detector 52
52
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD1 25
RCARD1 25
DCARD1 25
MANF_D
EXHAUST_2

SPARE
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD1 26
RCARD1 26
DCARD1 26
MANF_D
EXHAUST_2

Detector 25
25
PRESENT
REFV
MAXFT
```

```
SUPT
FILL_2
ACARD1 27
RCARD1 27
DCARD1 27
MANF_D
EXHAUST_2

Detector 24
24
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD1 28
RCARD1 28
DCARD1 28
MANF_D
EXHAUST_2

Detector 26
26
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD1 29
RCARD1 29
DCARD1 29
MANF_D
EXHAUST_2

/*=============
**  MANIFOLD E   **
*************/
Detector 50
50
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD2 0
RCARD2 0
DCARD2 0
```

```
MANF_E
EXHAUST_2

Detector 48
48
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD2 1
RCARD2 1
DCARD2 1
MANF_E
EXHAUST_2

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD2 2
RCARD2 2
DCARD2 2
MANF_E
NO_EXHAUST

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD2 3
RCARD2 3
DCARD2 3
MANF_E
NO_EXHAUST

Detector 23
23
PRESENT
REFV
MAXFT
```

```
SUPT
FILL_2
ACARD2 4
RCARD2 4
DCARD2 4
MANF_E
EXHAUST_2

Detector 30
30
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD2 5
RCARD2 5
DCARD2 5
MANF_E
EXHAUST_2

/*============
**  MANIFOLD F   **
************/
Detector 46
46
PRESENT
REFV
MAXFT
SUPT
FILL_1
ACARD2 16
RCARD2 16
DCARD2 8
MANF_F
EXHAUST_2

Detector 44
44
PRESENT
REFV
MAXFT
SUPT
FILL_1
ACARD2 17
RCARD2 17
DCARD2 9
```

```
MANF_F
EXHAUST_2

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_1
ACARD2 18
RCARD2 18
DCARD2 10
MANF_F
NO_EXHAUST

Detector 22
22
PRESENT
REFV
MAXFT
SUPT
FILL_1
ACARD2 19
RCARD2 19
DCARD2 11
MANF_F
EXHAUST_2

Detector 21
21
PRESENT
REFV
MAXFT
SUPT
FILL_1
ACARD2 20
RCARD2 20
DCARD2 12
MANF_F
EXHAUST_2

Detector 29
29
PRESENT
REFV
MAXFT
```

```
SUPT
FILL_1
ACARD2 21
RCARD2 21
DCARD2 13
MANF_F
EXHAUST_2

/*=============
**  MANIFOLD G   **
*************/
Detector 16
16
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 8
RCARD2 8
DCARD2 16
MANF_G
EXHAUST_3

Detector 17
17
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 9
RCARD2 9
DCARD2 17
MANF_G
EXHAUST_3

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 10
RCARD2 10
DCARD2 18
```

```
MANF_G
NO_EXHAUST

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 11
RCARD2 11
DCARD2 19
MANF_G
NO_EXHAUST

Detector 34
34
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 12
RCARD2 12
DCARD2 20
MANF_G
EXHAUST_3

Detector 32
32
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 13
RCARD2 13
DCARD2 21
MANF_G
EXHAUST_3

/*============
**  MANIFOLD H   **
************/
Detector 27
27
```

```
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 24
RCARD2 24
DCARD2 24
MANF_H
EXHAUST_3

Detector 18
18
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 25
RCARD2 25
DCARD2 25
MANF_H
EXHAUST_3

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 26
RCARD2 26
DCARD2 26
MANF_H
NO_EXHAUST

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 27
RCARD2 27
DCARD2 27
```

```
MANF_H
NO_EXHAUST

Detector 38
38
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 28
RCARD2 28
DCARD2 28
MANF_H
EXHAUST_3

Detector 36
36
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD2 29
RCARD2 29
DCARD2 29
MANF_H
EXHAUST_3

/*============
**  MANIFOLD I   **
************/
Detector 28
28
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 0
RCARD3 0
DCARD2 32
MANF_I
EXHAUST_3

Detector 20
20
```

```
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 1
RCARD3 1
DCARD2 33
MANF_I
EXHAUST_3

Detector 19
19
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 2
RCARD3 2
DCARD2 34
MANF_I
EXHAUST_3

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 3
RCARD3 3
DCARD2 35
MANF_I
NO_EXHAUST

Detector 42
42
PRESENT
REFV
MAXFT_L
SUPT
FILL_3
ACARD3 4
RCARD3 4
DCARD2 36
```

```
MANF_I
EXHAUST_3

Detector 40
40
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 5
RCARD3 5
DCARD2 37
MANF_I
EXHAUST_3

/*============
**  MANIFOLD J   **
*============*/
Detector 31
31
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD3 16
RCARD3 16
DCARD2 40
MANF_J
EXHAUST_4

Detector 33
33
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD3 17
RCARD3 17
DCARD2 41
MANF_J
EXHAUST_4

Spare
0
```

```
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD3 18
RCARD3 18
DCARD2 42
MANF_J
NO_EXHAUST

Detector  6
6
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD3 19
RCARD3 19
DCARD2 43
MANF_J
EXHAUST_4

Detector  7
7
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD3 20
RCARD3 20
DCARD2 44
MANF_J
EXHAUST_4

Detector  1
1
PRESENT
REFV
MAXFT
SUPT
FILL_2
ACARD3 21
RCARD3 21
DCARD2 45
```

```
MANF_J
EXHAUST_4

/*============
**  MANIFOLD K   **
************/
Detector 35
35
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 8
RCARD3 8
DCARD2 48
MANF_K
EXHAUST_4

Detector 37
37
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 9
RCARD3 9
DCARD2 49
MANF_K
EXHAUST_4

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 10
RCARD3 10
DCARD2 50
MANF_K
NO_EXHAUST

Spare
0
```

```
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 11
RCARD3 11
DCARD2 51
MANF_K
NO_EXHAUST

Detector  8
8
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 12
RCARD3 12
DCARD2 52
MANF_K
EXHAUST_4

Detector  2
2
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 13
RCARD3 13
DCARD2 53
MANF_K
EXHAUST_4

/*=============
**  MANIFOLD L   **
*============*/
Detector 39
39
PRESENT
REFV
MAXFT
SUPT
FILL_3
```

```
ACARD3 24
RCARD3 24
DCARD2 56
MANF_L
EXHAUST_4

Detector 41
41
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 25
RCARD3 25
DCARD2 57
MANF_L
EXHAUST_4

Spare
0
NOT_PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 26
RCARD3 26
DCARD2 58
MANF_L
NO_EXHAUST

Detector  9
9
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 27
RCARD3 27
DCARD2 59
MANF_L
EXHAUST_4

Detector 10
10
```

```
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 28
RCARD3 28
DCARD2 60
MANF_L
EXHAUST_4

Detector  3
3
PRESENT
REFV
MAXFT
SUPT
FILL_3
ACARD3 29
RCARD3 29
DCARD2 61
MANF_L
EXHAUST_4
/*===============================================*/
/* The MANIFOLDS */
/*===============================================*/
12 /* Number of manifolds */

#define MF_VREF  -60 /* Reference voltage (millivolts) */
#define MF_MAXFT 600.0 /* Max. allowed fill time (seconds) */
#define MF_SUPT 25.0 /* Closure delay time (seconds) */

/* Format of data for manifold:

   Card-address Channel-number (for control of inlet valve)
   Card-address Channel-number (for control of purge valve)
   Card-address Channel-number (for liquid sensor input)
   Reference voltage for sensor
   Maximum fill time in seconds
   Closure delay time in seconds

*/

/* Manifold A */

RCARD1 06
RCARD1 07
```

```
ACARD1 07
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_1

/* Manifold B */

RCARD1 22
RCARD1 23
ACARD1 23
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_1

/* Manifold C */

RCARD1 14
RCARD1 15
ACARD1 15
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_1

/* MANIFOLD D */

RCARD1 30
RCARD1 31
ACARD1 31
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_2

/* Manifold E */

RCARD2 06
RCARD2 07
ACARD2 07
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_2

/* Manifold F */
```

```
RCARD2 22
RCARD2 23
ACARD2 23
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_2

/* Manifold G */

RCARD2 14
RCARD2 15
ACARD2 15
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_3

/* MANIFOLD H */

RCARD2 30
RCARD2 31
ACARD2 31
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_3

/* Manifold I */

RCARD3 06
RCARD3 07
ACARD3 07
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_3

/* Manifold J */

RCARD3 22
RCARD3 23
ACARD3 23
MF_VREF
MF_MAXFT
MF_SUPT
```

```
EXHAUST_4

/* Manifold K */

RCARD3 14
RCARD3 15
ACARD3 15
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_4

/* MANIFOLD L */

RCARD3 30
RCARD3 31
ACARD3 31
MF_VREF
MF_MAXFT
MF_SUPT
EXHAUST_4

/* Exhaust valves */

4
RCARD1 2
RCARD1 26
RCARD2 10
RCARD3 18
```

Jomar Hønsi

Liverpool, 10th of January 1995

E-mail: jh@ns.ph.liv.ac.uk

Telephone: int + 51 794 3398