AF2 Software Upgrade Feasibility Study

AF2 Software Upgrade Feasibility Study

Stephen Goodsell

ING Catalogue No WHT-AF2-10

Date : 15/04/02 Ver: 0.1 Draft

1. Introduction

This report concludes the EMM approved two-week feasibility study commenced on 11th March 2002 and was completed on the 22nd March 2002 by S Goodsell and R Bassom of the ING.

The aim of the study was to examine the existing AUTOFIB-2 (AF2) micro software, assessing the feasibility of a software upgrade [1].

It was agreed that three different operating systems would be considered, this evolved into 7 different upgrade options.

??	Option 1a	Upgrade to VxWorks on 68k (VME)
??	Option 1b	Upgrade to VxWorks on PowerPC (VME)
??	Option 2a	Upgrade to Real-time Linux on PowerPC (VME)
??	Option 2b	Upgrade to Real-time Linux on x68 (cPCI)
??	Option 2c	Upgrade to Real-time Linux on x68 (Desktop - PCI)
??	Option 3a	Upgrade to OS-9 on 68k (VME)
??	Option 3b	Upgrade to OS-9 on PowerPC (VME)

This report starts with an instrument review, describing the existing control hardware and software of the system, expressing reasons to why an upgrade is desirable.

The software and control hardware upgrade research is then described. Assessing each of the three operating systems in terms of development tools, software portability, licensing and support. The control hardware research looks into different replacement cards for each option. This section reveals why each option was considered.

The report then discusses the cost, required effort and risk associated with each option, concluding with a section detailing the recommendations that come from the researched gained from this study.

2. Instrument Review

This review section, after a short instrument description, briefly examines the existing control hardware and software before explaining the reasons why a software upgrade is desirable.

A wealth of information is contained within the AF2 website [2]. It includes a page that lists all AF2 and relevant Wide-field Spectroscopy documentation [3]. Documents of particular use include the Autofib2/WYFFOS User Manual [4], the Autofib2 Technical Manual and the Autofib2 Control System Software Manual, which contain information on the usage, control hardware, and software respectively.

It should be noted that some sections in these documents need updating, especially information associated with the commissioning of small fibres and the decommissioning of large fibres.

AutoFib-2 is the WHT prime focus robotic fibre positioner. It was originally built within a collaboration between Durham and RGO. After commissioning the instrument was plagued by unreliability and consequently the robot and fibre module were re-engineered by ING staff.

When mounted on the telescope, AF2 is positioned behind the prime focus corrector. The fibres are positioned on a field plate by a robotic set of jaws with 3-d motion capability. The fibres are run along a spider and then down the side of the telescope finally entering the GHRIL cabin feeding the bench mounted spectrograph WYFFOS.

The instrument operates with a small fibre module. This contains 150 science fibres with 1.6 arcsec diameter (90 micron). The fibres are high-content OH fused silica made by Polymicro. This module also contains 10 new fiducial bundles, available for field acquisition and (auto)guiding. Each fiducial bundle (450 micron diameter) contains 10000 coherent fibres providing a rough imaging capability over an 8 arcsec round field.

2.1. Existing AF2 Control Hardware

The functions of each of the VME cards are briefly described in following sub sections. More details on the cards can be found in the individual cards user manuals whereas more information on their AF2 usage can be found in the Technical Manual [5].

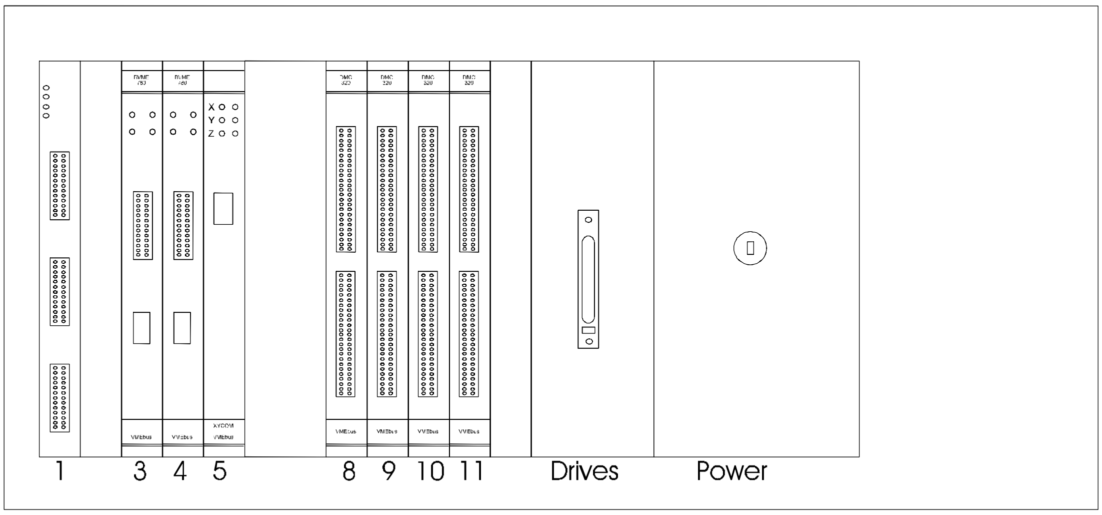


Figure 2.1 *Current AF2 VME crate*

The instrument micro computer is contained in a 7U 19” rack. In the rear of the crate is a power supply and a VMEbus 12 slot backplane. Figure 2.1 shows the front view of the crate and cards. The VME cards and IP cards are listed in table 2.1. All cable connections to the instrument, servo amplifier and monitor displays are made to the rear of the crate.

SLOT	MODEL	MANUFACTURE	FUNCTION
VME 0	TP32V (68030 processor)	Tadpole Technology	Single Board Computer
VME 1	Not used		
VME 2	BVME780	BVM Ltd	Frame Grabber/Graphics Card
VME 3	BVME780	BVM Ltd	Frame Grabber/Graphics Card
VME 4	Custom Board*	Durham University	Interconnection Facility
VME 5	Not used		
VME 6	Not used		
VME 7	DMC320	Galil	Motion Control Card
VME 8	DMC320	Galil	Motion Control Card
VME 9	DMC320	Galil	Motion Control Card
VME 10	DMC320	Galil	Motion Control Card
VME 11	VIPC310	Greenspring	IP Carrier Card
VME 12	Not used		
IP A	IP-Precision ADC	Greenspring	ADC
IP B	IP-Digital 24	Greenspring	Digital I/O

* This custom card isn’t a true VME card, but just takes power from the Backplane.

Table 2.1 *List on AF2’s VME and IP Cards*

2.1.1. Single Processor Boards

The instrument currently uses the TP32V 68030 Single Board Computer [6], from Tadpole Technology PLC.

There are 6 connections made to this cpu board.

1. SCSI connector
2. RS232 connector
3. Parallel printer connector
4. Hard/floppy disk boot sequence switch
5. Ethernet transceiver
6. Floppy disk drive connector

AF2 currently boots from a hard disk. It uses the Ethernet transceiver and the RS232 connector to communicate with the network.

2.1.2. Framegrabber / Graphics Cards

There are two BVME780 framers/graphics cards [7] on the VME crate. One acts as a frame grabber for the robot gripper TV system allowing the software to grab images of the fibres to determine their position. This card has two external connections, a video input from the gripper TV camera and a processed video output in order to display what the frame grabber and software are doing with the image of the fibres.

The second BVME780 is used as a graphics card for the engineering mimic display. This card has five video connections, the inspection TV camera inside Autofib-2 is connected to the video input and the output RGB and Sync are connected to a colour monitor.

2.1.3. Interconnectivity Card

This custom designed and manufactured board that simply provides an interconnect facility for all the other cards and the multipole connectors to the outside world. Although this card resides in the VME crate it is not a true VME bus card and simply takes power from the backplane with no other connections to the VME bus being made via the backplane.

2.1.4. Motion Control Cards

There are four Galil DMC320 [8] motion control cards connected to the crate. Each of these cards is capable of controlling two motor axes. The cards contain the following motors and encoders:

?? X axis motor and rotary encoder
?? X axis linear encoder
?? Y axis motor and rotary encoder
?? Y axis linear encoder
?? Z axis motor and rotary encoder
?? Theta axis motor and rotary encoder
?? Off axis probe motor and rotary encoder

Each motor or encoder axis uses a 26 way ribbon cable to connect to the interface board. This cable provides all the motor command, encoder signals, limit and home position switch signals.

2.1.5. IP Carrier Card

The last VME card to be connected to the backplane is the Greenspring VIPC310 dual Industry Pack (IP) carrier board. [9] This board contains two IP cards that give ADC and digital I/O capabilities.

2.1.6. Digital Input/Output IP Card

The IP-Digital 24 IP card [10] is seated in the first IP slot. All the control and the status of the instrument is provided by this card as tabulated in table 2.2, below. Channels 1-8 are used for control (output), whereas channels 9-16 are used as status (input).

Channel	Control Line	Channel	Status Meaning
1	Instrument Power on/off	9	Instrument power status
2	Motor Relay on/off	10	Air pressure status
3	Gripper open/close	11	Motor relay status
4	Instrument inspection lights on/off	12	Fibre Module proximity switch A
5	Limit switch override control	13	Fibre Module proximity switch B
6	Guide Fibre back illumination	14	Servo amplifier status
7	Spec. fibre back illumination (n u)	15	Spec fibre back illumination status (n u)
8	(not used)	16	Guide fibre back illumination status

Table 2.2 *IP-Digital 24 IP card channel descriptions*

2.1.7. Analogue Digital Converters (ADC) IP Card

The second IP-Precision ADC IP card [11] provides 20 channels of 12-bit analogue to digital conversion. Only one channel is used to measure a voltage, which gives the temperature of the instrument.

2.2. Existing AF2 Software

This section outlines the instruments low-level software running on the TP32V 68030 Single Board Computer. Rather than attempt to describe everything, it concentrates on issues that would be affected in a software upgrade, mainly code portability. It breaks the software into structure, real-time operation, and source code before finally considering other AF2 software.

The latest version of the software can be found on the AF2 micro's hard drive, in the /h0/autofib/ directory and is comprehensively documented [12].

2.2.1 Structure

AF2 currently uses OS-9/68K v2.4 operating system; this is a very old and obsolete version that doesn't support Y2K. The latest OS-9/68K operating system is v3.2.

There are currently 4 user accounts on the system. The Autofib account contains the operational software. The other 3 are currently for development purposes. Each account has its own root directory, username and password. The Autofib home directory is /h0/autofib/. It contains 12 directories that can be seen below in Figure 2.1.

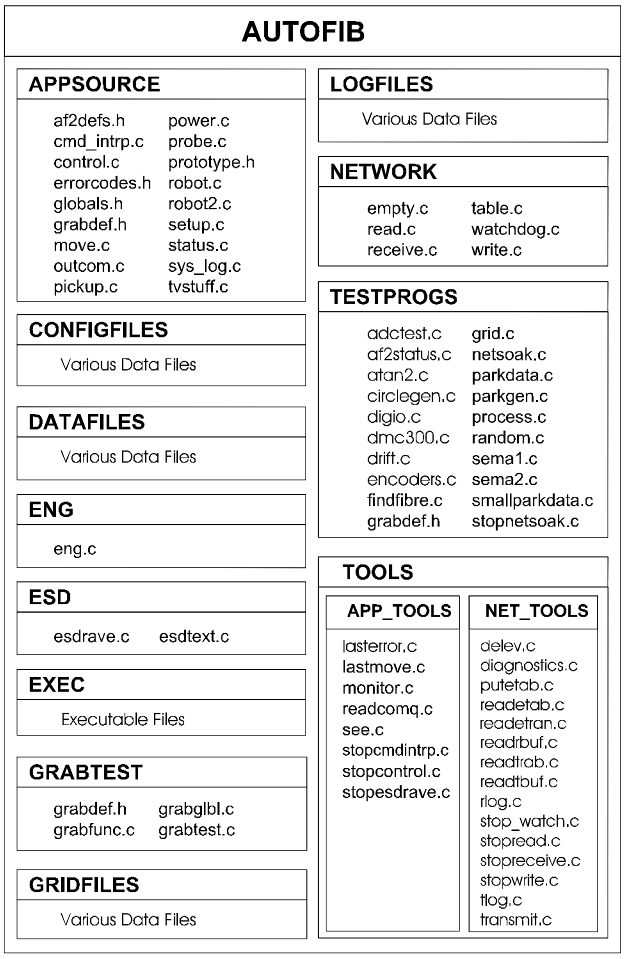


Figure 2.1 *AUTOFIB directory Structure*

Only the files that are important in terms of an upgrade are displayed. Other files such as data files, makefiles, scripts and executables are not shown. Most of these files are generated through compiling, running software or logging. The scripts, that reside in the /h0/autofib/ directory can easily be transferred or reproduced.

2.2.2 Real-time Operation

The Autofib-2 control system consists of a number of tasks running in the background on the instrument micro computer. Four tasks provide the Utility Network interface between the instrument and the outside world. Two tasks provide for instrument status reporting and control. A final task provides an engineering mimic display. The tasks are shown in table 2.3.

Task	**Type**	**Description**
Control	Control	Main Control Task
Cmdintrp	Control	Command Interpreter
Read	Network	Network Layer Read Task
Receive	Network	Network Layer Receive Task
Write	Network	Network Layer Write Task
Watchdog	Network	Network Layer Watchdog Task
Esdrave	Engineering	Engineering Mimic
Shell	OS-9	Command Line Shell
Procs	OS-9	Procs process displaying the information

Table 2.3 *AF2 OS-9 Software Tasks*

Communication between the various tasks is by two methods, software interrupts allow the immediate passing of information between tasks, in OS-9 these are called signals. The majority of information is passed through shared memory, the tasks check a semaphore value called an event to check that the memory resource is free. The format of the information in the shared memory areas is given in Appendix B of the Autofib2 Control System Software Manual, [12].

In operation a script is run, the script firstly runs a program that executes a ‘table’ program which crates all the datamodules needed. The script then spawns the watchdog, write, receive, read, control, cmd_intrp and esdrave in sequence setting different priority levels for each of the tasks.

At this stage the software is in a state where the user can send commands remotely. To send commands locally the user has to run the ‘eng’ program on the crate. This will allow the user to issue commands using a command line. The AF2 commands are well documented on the website and in the user manual.

2.2.3. Source Code

This part of the review looks at each task’s source code, estimating the source code’s portability. Each task is broken down into functions. Each function was examined to see if it contained any non-AF2 function calls that aren’t part of any ANSI C library.

Non-AF2 function calls are defined as OS-9 specific calls and calls using external libraries. It was found that they were 4 types of non-AF2 function calls, OS-9 operating system calls, OS-9 datamodule calls, OS-9 signal calls and finally BVME780 calls. Appendix A looks at each task in more detail, whereas this section describes a summary.

Along with the AF2 tasks, two executable programs were also examined, the table program, which creates the datamodules, and the eng program, which allows the user to issues commands locally.

The control task is the biggest task by far. It consists of all the .c files in the appsource directory with the exception of cmd_intrp.c. Whereas the cmdtintrp task is created from just the cmd_intrp.c file. Each of network tasks are made from .c files of the same name in the network directory whereas the esdrave task is created from the esdrave.c and esdtext.c files which reside in the esd directory.

The number of function contained within each task and executable is displayed in table 2.4. It also contains the number of functions for each task that contains non-AF2 function calls.

Software		No Functions	No Functions containing non-AF2 calls
Tasks	control	102	13
	cmdintrp	17	5
	esdrave	16	16
	read	6	4
	receive	11	5
	watchdog	4	3
	write	8	6
Executables	table.c	1	1
	eng.c	11	3
TOTAL		**176**	**56**

Table 2.4 *Table displaying portable functions*

On closer inspection, some of the functions that are contained within tasks are duplicate or very similar to a function of another task, this is to say to terms of portability they are equal. This would reduce portability even further. A total of 13 of the 53 functions have been duplicated, leaving 40 functions that are unique. This gives a final estimation of the code being 77% portable.

The non-AF2 function calls themselves are distributed between tasks. A total of 8 datamodule function calls are used, with most being present in each task. There are 3 OS-9 system function calls used in the control and cmdintrp tasks. 3 OS-9 signal function calls in the 4 network tasks and finally 20 BVME780 function calls in the control and esd tasks.

The tasks include functions that write to memory addresses. This is how the software controls and reads status from the motion controller cards, the ADC card and the digital I/O card. They don't use non-AF2 function calls.

2.2.4 Other Software

It should be noted that there is other software associated with the AF2 system. There is the field configuration software, the higher level ICL software and the data reduction software. A software upgrade would have no impact on the field configuration software and data reduction software, therefore these pieces of software needn't be considered. The ICL software needs to be considered as a change of communication protocol or command protocol within the micro software would have serious consequences.

2.3. Reasons for Upgrade

Reasons for a software upgrade were originally documented in the case for this study [1]. This section re-examines these reasons.

The Visiting Group strongly endorsed the ING proposal to concentrate on wide-field multi-object spectroscopy, among other things for the future.

'AUTOFIB/WYFFOS with the current upgrades will be competitive for at least the next 4 years. **The input of the ING project staff is critical to the success of this overall project.**' – Executive Summary of the Visiting Committee 2001. [13]

The ING restructure plan [14] shows AF2 being used to at least 2007. With this in mind it is desirable that the instrument can be continually improved to satisfy the demands of the astronomical community. A major software upgrade should only be done once, with small operation system upgrades until the instrument is decommissioned.

This instrument software should be

?? Maintainable
?? Upgradeable
?? Developable

Maintainable

Currently the software is very inefficient to maintain. With the reduction of staff in the next few years the ING will struggle to afford this. The instrument is the only instrument using OS-9, this leads to unwanted effort training people on an unfamiliar and non-standard system. The inefficiencies in using this current version of OS-9 can't be overstated.

Editing code is presently very inefficient. Currently software engineers move individual files across to a UNIX environment to use UNIX editing tools. The files are transferred backwards and forwards using ftp.

The micro software isn't in a CVS repository; currently the whole disk is being backed up. The CVS repository isn't used because of the effort required in understanding how to reconstruct everything from scratch on an OS-9 system. These inefficiencies all add up resulting in lots of wasted effort.

Upgradeable

The version of the operating system is obsolete and can't simply be upgraded. It is unsupported and not Y2K compliant. Upgrading the instruments operation system shouldn't require substantial effort. It should have been upgraded continuously in the past.

As it is not upgradeable, there is no chance of solving the biggest constant operational problem of the crate falling off the network [15]; Ian Lewis first documented this problem in 1996. [12]

Developable

The developments that can be presently done on the system are very limited. It is true to say that small software upgrades are feasible such as the changes required for small fibres. The majority of this was changing variables that already existed.

The biggest problem is the lack of development tools available. The only tools that are available are the tools and test programs that were written for the instrument along with some unusable editors.

There are no real-time development tools. Therefore it is currently impossible to optimise the system. The absence of tools makes developing software magnitudes more harder than it needs be. There are many advantages to having a good set of development tools, especially when developing for a real time system. Tools will make editing and debugging code much more efficient allowing code development and instrument enhancements possible.

Currently it isn't realistic to add extra mechanisms or optimise the instrument's performance.

Without going into detail, development tools would allow the following, which would increase operational efficiency;

?? Optimisation
  o Fin/Fibre misalignment compensation
  o Improved centroids routine
  o Code optimisation
?? Mechanisms
  o Controlled Backillumination positoning mechanism (linear device)
  o Automatic fiducial enlargement (rotatory device)

3. Upgrade Research

The material that this study covered was suggested in it's case [1]. The recommended focus of the study was to research into 3 different operating systems, VxWorks, Real-time Linux and OS-9.

After examining the requirements and collecting some initial information, it was decided that the study should concentrate on the 7 following different options involving the 3 suggested operating systems. These seven options are;

?? Option 1a	Upgrade to VxWorks on 68k (VME)
?? Option 1b	Upgrade to VxWorks on PowerPC (VME)
?? Option 2a	Upgrade to Real-time Linux on PowerPC (VME)
?? Option 2b	Upgrade to Real-time Linux on x68 (cPCI)
?? Option 2c	Upgrade to Real-time Linux on x68 (Desktop - PCI)
?? Option 3a	Upgrade to OS-9 on 68k (VME)
?? Option 3b	Upgrade to OS-9 on PowerPC (VME)

The documented research is split into 2 sub-sections. The first concentrates on the software issues, taking a look at the differences between each operating system, containing a description of what they have to offer. The other subsection discusses relevant required changes in the AF2 control hardware.

3.1. AF2 Software Research

VxWorks, Real-time Linux and OS-9 are the three operating systems that are considered in the upgrade options. The software section starts with a description of the different types of operating system. This is followed with each operating system being described in terms of features, development tools, licensing and support.

3.1.1 Introduction to Operating Systems

Before examining and comparing these systems it is important to understand why AF2 must use a real-time operating system.

There are three types of operating system.

?? Non real-time
?? Soft real-time
?? Hard real-time

Popular desktop operating systems such as Windows, Unix and standard Linux are seen as non real-time. They each have kernel that disables interrupts for long periods of time [16]. This is what makes standard Linux a non-real-time OS. It is important for AF2 that the software doesn't hang for undeterminable amounts of time or indeed crash during operation, concluding that a real-time operating system is needed.

A real-time operating system is one which can keep up with what's happening in its environment. It relates to the importance of deadlines. The definition of what makes operating system hard or soft real-time is a little vague. One definition is that a hard system is needed if missed deadlines means failure of the system, whereas for a soft system, missed deadlines means a degraded system. [17]

For AF2 it is vital that the instrument doesn't fail, as failure leads to downtime. How missed software deadlines affects the instrument is something that would become clear in examining the software. For example, it would be acceptable if a missed deadline causes a few milliseconds delay in moving a stepper motor.

It is hard to find out without having the real-time development tools that the operating system presently lacks. Without a clear definition between a soft and hard real-time operating system, and without development tools it is hard to know for sure if the present AF2 software requires a hard real-time operating system.

AF2 should be a soft real-time instrument as a missed deadline should cause the system to still work, as any part of it's operation isn't time critical. Whereas, NAOMI would definitely need a hard real time operating system, as if a deadline isn't met when the loop is closed then the system will fail.

OS-9 and VxWorks are hard real-time operating system, whereas some versions of Real-time Linux are soft real-time. Porting the software over to a soft real-time operating system would introduce risk. The risk should be small if Real-time Linux was used, as there would be at least a 10- fold increase in processor speed.

3.1.2. OS-9

OS-9 is developed by Microware Systems [18] who have since been acquired by RadiSys Corporation [19]. It is a well-established operating system.

OS-9 is available for a wide range of processors and boards. The processors family covered are 68k, ARM, IXP1200, MIPS, PowerPC, StrongARM, SuperH and the X86/Pentium family.

The latest package contains version 3.2 of the operating system, drivers, development board solutions, development environment and middleware. The version of OS-9 for the 68k [20] and the PowerPC [21] contain slightly different features.

The operating system basic components consist of a kernel, I/O manager, Pipe file manager, Sequential Character File manager, Sequential Block File manager, Ms-Dos format file manager and system security module. It network components include Stacked Protocol File manager, LAN Communication Pak with Ethernet/Internet support, TCP/IP, UDP and PPP prot. stack. The operating system is documented in the user manuals, the OS-9 technical reference manual is particularly useful. [22]

Development Tools

The latest version of OS-9 has a firm Integrated Development Environment (IDE) for Windows, Hawk IDE [23]. The suite of development tools consist of Powerful Editor (Codewright), Project Manager (VCCS), Optimised Ultra C/C++ Compiler, Target Debugger and Target Profiler. These tools are what are expected to comfortably write, compile and debug code for a real-time operating system.

The package that you buy, Board Level Support (BLS), includes the OS-9 v3.2 operating system, Hawk IDE and BSP is dependant on the VME board that you are developing for.

Software Portability

The OS-9/68k is not exactly the same compared to OS-9/PPC. OS-9/68k is almost completely portable and should be relatively straightforward. Both would require the existing source code to be ported from the native environment to Hawk (cross environment) and recompiled.

For the 68k all the non-AF2 function calls revealed in section 2.2.3 should work, at the most require slight changes. The PowerPC version of OS-9 makes the software a little less portable. The drivers for the frame grabber card may need modifications, there would need to be some memory-mapping configuration needed. The OS-9 specific function calls would still work. [24]

Licensing

The license would be a fixed cost regardless of which processor was used. Extra money would have to be paid, if more than one person developed (extra seat) or if more than one processor was to be used.

Support

Microware offer various support packages [25]. The package most suited to the needs of AF2 would be the standard support package;

One engineer and one designated backup engineer are entitled to one year of unlimited phone, e-mail, and web-based support for installation, configuration, usage issues, "work-arounds," patches, maintenance releases and product updates. The initial response time for the Standard Package is within 2 business days.

3.1.3. VxWorks

VxWorks is produced by Wind River Systems Inc. [26]. Currently there are 2 separate strands of the VxWorks operating system, VxWorks 5.x [27] and the recently released VxWorks AE [28]. Both products lines are being developed, only VxWorks 5.x is considered because it is the most relevant to both the upgrade and the ING, it doesn’t support 68k family of processors.

VxWorks 5.x covers the PowerPC, ColdFire, 68K, M-CORE, x86/Pentium, StrongARM, Xscale, MIPS, ARM and SuperH architecture.

The latest version is VxWorks 5.4. VxWorks, the run-time component of the Tornado II embedded development platform, is the most widely adopted real-time operating system in the embedded industry.

VxWorks comprises the core capabilities of it's microkernel with advanced networking support, powerful file system and I/O management, and C++ and other standard run-time support. All is well documented within the programmer's guide. [29]

Development Tools

Tornado II [30] [31], comprises a comprehensive suite of core and optional cross-development tools and utilities.

Standard Tornado II tools:

?? ***WindSh:*** The WindSh shell interface allows users to interact with all target facilities. Unlike other "shells", the Tornado shell can interpret and execute almost all C-language expressions, including calls to functions and references to variables whose names are found in the system symbol table. Interpreted C statements give an easy-to-use interface to the target environment.

?? ***Browser***: The Tornado browser is a graphical companion to the Tornado shell, presenting information symbolically whenever possible. The browser gives the overall state of the system and allows developers to launch dedicated displays that monitor the state of target operating-system objects, such as tasks, semaphores, message queues, memory partitions and watchdog timers.

?? ***WindNavigator***: The WindNavigator Multilanguage browsing tool enables developers to dramatically reduce the time to evaluate existing C and C++ source code, even if the code is incomplete or erroneous. With WindNavigator, developers can see the relationship between objects and functions and can easily build programs using existing, proven modules.

Other standard Tornado II tools include CrossWind, MemScope, Project, Rshell and LmServer. All of these tools are described in detail in the Tornado User's Guide [32]

The following Wind River's WindPower tools are optional.

?? ***WindView***: The WindView diagnostic and analysis tool provides detailed visibility into application software as it runs on the target hardware. [33]

?? ***VxSim***: The VxSim prototyping and simulation tool enables developers to create prototype applications – including networking and multiprocessor-based designs – before the actual target hardware becomes available. VxSim also permits developers to test a large portion of application software early in the development cycle, when errors are less costly to correct.

?? ***ScopePak***: ScopePak includes the *StethoScope* [34] and *TraceScope* [35] real-time data visualization, profiling, and debugging tools that let developers analyze an application while it's running.

?? ***PerformancePak***: The PerformancePak includes *ProfileScope* [36], which provides detailed, function-by-function analyses of tasks running within an application and MemScope, which helps developers control memory usage.

?? ***CodeTEST*** *for Tornado*: This software-only version of CodeTEST is an easy-to-use, cost effective tool for analysing embedded program as they run in a target system. It is available in two separate modules:
  - **CodeTEST/Coverage** pinpoints untested areas of code and provides dynamic views of the system.
  - **CodeTEST/Memory** permits developers to observe dynamic memory levels as they occur.

?? ***Visual SlickEdit*** – Tornado Edition: The full-featured editing capabilities of Visual SlickEdit are combined with special support for the Tornado development environment.

?? ***Wind Foundation Classes***: VxWorks Wrapper classes and Toolsh++ from Rogue Wave Software support object-oriented design and help accelerate application development.

?? ***Look***! For Tornado: This C++ visualization and debugging tool graphically explores a C++ program as it executes.

A complete range of comprehensively documented development tools is available for VxWorks.

Software Portability

Comprehensive documentation exist on intertask communications [29]. VxWorks is capable of shared data structures, mutual exclusion, semaphores, message queues, pipes, network intertask communication and signals. It is capable of recreating the shared memory and signals that OS-9 use. There are operating system calls similar to the OS-9 calls that are used in the AF2 software.

The graphics cards device driver wouldn't be compatible and would have to written for VxWorks. The memory map would have to be configured so it would be possible to write to the existing cards, no matter if a 68k processor or a PowerPC processor was used.

Current Licensing

The pricing of VxWorks and licensing is very complicated. The licensing depends on number of sites, number of projects, target microprocessor architecture family, processor family, how many targets and how many developers you have. The licensing is fully described in Appendix B with examples.

Support

VxWorks offers maintenance contracts. With valid maintenance it is possible to access WindSurf, a web based maintenance site. There are different forms of support that are described in the 'Customer Support' User Guide. [37]

3.1.4. Real-time Linux

Unfortunately time did not allow for a comprehensive research into Real-time Linux. There are many different flavours of Real-time Linux, summarised in "The Real-time Linux Software Quick Reference Guide". [38]

There are a number of companies distributing commercial versions of Linux [39-42], while a number of organisations distribute one of two main open-source versions. The two open source versions of Real-Time Linux are RTAI and RTLinux. Both of these are hard-real time and are distributed by many organisations.

RTLinux is a hard real-time operating system that runs Linux as its lowest priority execution thread. The Linux thread is made completely preemptible so that realtime threads and interrupt handlers are never delayed by non-real-time operations. Real-time threads in RTLinux can communicate with Linux processes via shared memory or a file-like interface, so real-time applications can make use of all the powerful, non-real-time services of Linux. For example, it is easy to write a Perl script that displays data in Xwindows, responds to commands delivered over a network, and collects data from a real-time task. [43]

RTAI is a comprehensive Real Time Application Interface that is usable both for uniprocessors (UP) and for symmetric multi processors (SMP), that allows the use Linux kernel 2.2.xx for many "hard real-time" Linux applications.

The latest version of RTAI combines a Real-Time Hardware Abstraction Layer (RTHAL) with a Real-Time Application Interface (RTAI), to make Linux usable for hard real-time applications.

Development Tools

Each commercial company either has it's own set of cross development tools or board support package that would only work with the their version of Real-time Linux. This meant that although Linux was free, their development tools aren't.

There are a few free tools available, but none as extensive as the tools offered by VxWorks or even OS-9.

Tools include a real-time scheduler, a trace toolkit, a RTOS simulator and a project characterisation tool. Each of these is maintained by different organisations.

Software Portability

Porting the software over would be possible but would be very tricky without the right tools and support. To take advantage of Linux the code would have to be split into its real-time and non real-time components, making things messy when you are trying to port code. More investigation is needed.

Current Licensing

There are no licensing issues if an open-source version is used. If a commercial version were used then a licensing fee could be charged.

Support

Again commercial companies offer support and support contracts. Open source distributions rely on support from the Linux community and the support is free.

3.2. AF2 Control Hardware Research

This sub-section looks at describing the control hardware required for each upgrade option. The two-week study didn't provide enough time to find the best possible solution for each option, only that a solution for each option exists.

This section looks at the bus, processor, framegrabber, motion controllers and input/output cards in turn, suggesting solutions. All the control hardware that needs to be replaced is described with the exception of cables.

3.2.1. Buses

There are many buses available to the embedded world, VME, VME64X, VME320, VXI, cPCI and CTEL. This study considered the VME [44] and the cPCI buses [45].

This choice was made because all the instruments at the ING that has VxWorks as an operating system uses a VME backplane. AF2, the only instrument that uses OS-9, also uses the VME bus for inter-card communication.

If OS-9 or VxWorks were used then it would make sense to use a standard VME backplane to connect the control hardware. Real-time Linux can also be installed on a VME processor card, placed in a VME backplane.

If a VME backplane were to be used then it would be sensible to use the existing AF2 VME backplane. The current backplane has 12 slots, which is more than enough for any of the VME based options.

The other backplane option involves using a CompactPCI (cPCI) backplane. CompactPCI is a modern, very high performance industrial bus based on the standard PCI electrical specification, on a rugged 3U or 6U Eurocard packaging. Unlike its desktop cousin, the CompactPCI adapters use a high quality 2mm metric pin and socket connector and can be front loaded into a rack mount system. CompactPCI cards are also hot swappable, meaning they can be changed without powering down the system.

The cPCI bus would need to have at least 5 cPCI slots assuming that the processor card has an Ethernet interface and an AGP device, see section 3.2.2. This option was considered because it was thought that cPCI cards are cheaper than VME cards and they are more likely to have Linux drivers written for them, while still being ruggedised. Unfortunately, as shall be seen, the cPCI isn't much cheaper, or more 'Linux friendly' than the VME Real-time Linux option.

PCI was also considered for Real-time Linux, mainly because of 6dF [46]. 6dF is a similar instrument to AF2 that resides at the AAO. The control hardware consists of a desktop PC running Linux and using PCI cards [47]. PCI buses aren't normally considered in the embedded world because they are ruggedised.

The reasons described led to the different choices of backplanes found in the options.

3.2.2. Single Processor Boards

The existing processor card is obsolete and isn't supported by the latest versions of OS-9 and VxWorks [48].

Experiments were performed to estimate the affects of using a faster processor. The experiments are described in Appendix C. The results predict that a set-up would be up to 40% quicker if we used a top end processor and made some code modifications. This gave the incentive to research into faster processors than the 68k.

Rather to look at every single architecture family and every single manufacture of CPU boards, it was sensible to look at Motorola boards, as they are a widely used manufacturer whose information is easily accessible. The 68k family of processors was compared with the PowerPC family of processors for the VME bus. The 68k processor is currently used in the ING instrument's control hardware whereas other observatories such as Gemini are using the faster PowerPC's.

Motorola is still selling the 68k processor because of organisations that don't need to upgrade that hardware to faster boards. As the processor isn't in as much demand as the PowerPC, they are more expensive.

The x68 Pentium 3 processor was examined for the PCI and cPCI Linux options. This was because the x68 families contain the fastest commercial processors in the world.

VME 68k Embedding Controllers and Single Board Computers

There are 5 VME boards in this range, the MVME162, MVME172, MVME147, MVME167 and the MVME177. [49-53]

The 68k chips have been around for over 10 years. The present AF2 board contains a 68030 processor running at 32 MHz. The Motorola processors run at speeds from 16MHz to 60MHz.

The ING already use MVME147 and MVME167, if deciding to develop for the 68k processor it makes sense to use an ING spare. Unfortunately the VxWorks development license the ING currently have is for a 68020 microprocessor whereas the present available Motorola boards contain either the 68030, 68040 or 68060 processor, this implies that the ING would have to pay for a new development license.

This isn't really an issue with OS-9, as the ING would have to get a new license anyway.

VME PowerPC SPB

There are 8 different Motorola VME boards that contain processor chips. This range consists of the MVME2100, MVME2300, MVME2400, MVME2600, MVME2700, MVME3600, MVME4600 and the MVME5100. [54-60]

The PowerPC processor currently ranges from 100MHz to 500MHz. The top end PowerPC was released in April 2002. If this option is decided then addition research in choosing the best PowerPC is required.

In addition to the PowerPC, a transition module would be needed that contains serial, parallel and Ethernet ports.

New licenses would be required for both the VxWorks and OS-9 options.

cPCI Pentium SBC

The speeds of the Pentium 3 processors range from 500 MHz to 1 GHz. The top processor would increase the speed by over 30 times.

Both the Motorola CPN5360 card [61], Trenton Technologies CPLE [62] and Trenton Technologies CPBI [63] were considered. Each card would have a transition modules that contained ports. Each card was found suitable of hosting a hard real-time version of Real-Time Linux.

Desktop Pentium with PCI

There are many desktops that are suitable, so only one was looked at, this happened to be machine from Arcom Control Systems, it contains a Pentium 3 processor and a 12 PCI slots along with hard drive, floppy drive, CD Rom and other input/output ports. [64]

3.2.3. Framegrabber / Graphics Cards

The BVME Framegrabber cards are now obsolete although BVM Ltd have informed the ING that they are willing to remanufacture the card if a minimum of 5 is ordered [65]. The best solution for the OS-9 would be to use the existing cards.

There was a couple of suitable framegrabber cards found. The first choice card was the VME DFG2 framegrabber card from Dynatem Inc.

Another framegrabber card is the PMC ‘Snapper’ Framegrabber card [66]. It is possible to get a PMC carrier card for the cPCI option to house the ‘Snapper’ framegrabber card. VxWorks drivers are available for each card.

There are many PCI Framegrabber cards that have Linux drivers, one such example is the DT3155 framegrabber card [67]. It comes with a Window’s software development kit but Linux drivers must exist as it is used in 6dF [46]

3.2.4. Motion Control Cards

The original motion control can be used in the VME options because as we have seen, the software writes to registers. These cards are obsolete but the makers, Galil, now sell the next generation DMC-1348 cards [68]. The next generation cards communicate in the same way as our existing cards, so they would be an ideal replacement. [69]

An ideal motion control card for the cPCI Linux option would be the DMC-1640 cPCI based motion control card. Whereas the PCI option would be the DMC-1842, all contain the same functions, but are for the respective backplane. [68]

3.2.5. IP Carrier Cards

Greenspring, the manufacture of the original AF2 IP carrier card are now called SBS Technologies. The IP Carrier Card VIPC310 is still being sold. It isn't obsolete and could be used for the VME systems.

A cPCI IP carrier card, such as the CIPC Intelligent DSP Based Dual Industry Pack Carrier from Dynatem Inc. would be needed. [70] This would allow the cPCI option to use the existing cards as discussed below.

The PCI Linux option wouldn't need an IP Carrier Card.

3.2.6. Analogue Digital Converters (ADC) Cards

The IP card used in the existing system is obsolete yet the IP-Unidig supplied by SBS offers a replacement [71]. As the software currently just writes to registers, the most sensible thing would be to use the existing card for the VME systems, making sure that the replacement could easily be substituted, it should only be purchased if one of the two ING cards malfunctions.

All VME options and the cPCI option can use this card whereas the PCI Linux option could use the DT302 data acquisition cards, produced by Data Translation. [72] This card contains both analogue input lines and digital input/output lines and is used in 6dF. [46]

3.2.7. Digital Input/Output Cards

Again the current IP DI/O card is obsolete. The IP-OPTOAD12 is a similar IP card [73], which again could be used as a replacement. The original card could be used and the new card only purchase if one of the two ING cards malfunctions.

As mentioned before we presently write into addresses for both cards therefore we wouldn't need to obtain any drivers for the card. This would be used in both the VME and cPCI systems, whereas the above-mentioned DT302 would be used in the PCI system.

3.2.8. Interconnectivity Card

This card would either remain the same in the case of the VME backplane or be modified for the cPCI option. How the cables would come together in the PCI option would be something that would need research.

4. Cost, Effort & Risk

This section discusses the cost, required effort and risk associated with each option.

Just to recap the 7 different upgrade options are:

?? Option 1a	Upgrade to VxWorks on 68k (VME)
?? Option 1b	Upgrade to VxWorks on PowerPC (VME)
?? Option 2a	Upgrade to Real-time Linux on PowerPC (VME)
?? Option 2b	Upgrade to Real-time Linux on x68 (cPCI)
?? Option 2c	Upgrade to Real-time Linux on x68 (Desktop - PCI)
?? Option 3a	Upgrade to OS-9 on 68k (VME)
?? Option 3b	Upgrade to OS-9 on PowerPC (VME)

Costing for each option is given. This includes, the price to purchase the required control hardware, spares, software, software licenses and annual software maintenance. This doesn't include the cost of effort or involve any contingency. All figures provided in this document are in Euros, all prices obtained are summarised in Appendix D. In this section all prices have been rounded up to the nearest 10 euros, they don't include VAT or any shipping charges.

For the Real-time Linux options, it has been assumed that rather than purchase any commercial versions or tools, the ING would obtain the free open source version.

After considering the latency of risk elimination research, obtaining the hardware, establishing a development environment, transferring and developing the code, bench testing and final commissioning, the time taken to complete the project for each option is predicted. It is important to realise that these are estimates, based on this research and the author's knowledge and familiarity of the instrument. The timescales would be slightly longer for someone else being supervised, while considerably longer if someone else working in isolation. What is of particular interest is the relevant project length for each option. The estimated time is the time required for one person to work full time on the project.

Risks associated with each option are commented. The feasibility study did not allow enough time to totally eliminate risk for each option. Further risk elimination research would be required on choosing an option. All technical risks would be eliminated before purchasing hardware. A risk factor for each option has been issued. It relates to the risk on schedule, completing the project on time and the probability of encountering unseen problems. It is important to note that no option has a high risk factor. All options are feasible.

4.1. Option 1a Upgrade to VxWorks on 68k (VME)

This option would get the ING away from the OS-9 operating system, using a standard ING real-time operating system. It has an excellent supported set of development tools. It doesn't give any gains in processing speed, using the same technology that was available 10 years ago. It would take a relatively short time to finish this low risk project.

Cost

Table 4.1 contains the prices (in euros) of the control hardware, software and software licenses that the ING would be required to purchase for this option.

Item	**Cost**
VME MVME147 Board	7,840
VME 'DFG2' Framegrabber Card	2,270
Software and Licenses	10,900
Total	**21,010**

Table 4.1 *Hardware and Software Costing*

The software and hardware costs are roughly equal. Software maintenance each year would come to 8,340 euros. The cost of spares for this option would come to 10,110 euros.

Effort

The ING already have everything they need to start the project. An existing VME card that contains an old processor could be used for development until the new board arrived. The ING already have VxWorks and a set of development tools. The most amount of effort would be associated with porting the software.

Estimated Time **6 months**

Risk

There is low risk with this option, using an operating system, tools and environment that ING developers already have experience with. One risk would be making sure the framegrabber card worked with the camera. This would be eliminated before starting the project.

Risk Factor **Low**

4.2. Option 1b Upgrade to VxWorks on PowerPC (VME)

This option would get the ING away from the OS-9 operating system, using a standard ING real-time operating system. It has an excellent supported set of development tools. It gives more than a 10x increase in processor speed. It would take some time to finish this low risk project.

Cost

Table 4.2 contains the prices (in euros) of the control hardware, software and software licenses that the ING would be required to purchase for this option.

Items	**Cost**
VME MVME5100 500MHz PowerPC	6,700
VME MVME761 Transistion Module	350
VME 'DFG2' Framegrabber Card	2,270
Software License	28,890
Total	**38,210**

Table 4.2 *Hardware and Software Costing*

The software costs 3 times as much as the hardware needed. Software maintenance each year would come to 9,760 euros. The cost of spares for this option would come to 9,320 euros.

Effort

It would take 5-6 weeks for the PowerPC to arrive after being ordered. Some of this time could be used doing further research and experiment porting the AF2 software onto an old processor board containing VxWorks. The ING already have VxWorks and a set of development tools. The most amount of effort would be associated with porting the software.

Estimated Time **6 months**

Risk

There is low risk with this option, using an operating system, tools and environment that ING developers already have experience with. One risk would be making sure the framegrabber card worked with the camera. This would be eliminated before starting the project.

Risk Factor **Low**

4.3. Option 2a Upgrade to Real-time Linux on PowerPC (VME)

This option would get the ING away from the OS-9 operating system but introduce a previously unused real-time operating system. A set of good open-source development tools may exist but wasn't found. This option gives more than a 10x increase in processor speed. It would take a relatively long time to finish this medium risk project.

Cost

Table 4.3 contains the prices (in euros) of the control hardware, software and software licenses that the ING would be required to purchase for this option.

Items	Cost
VME MVME5100 500MHz PowerPC	6,700
VME MVME761 Transistion Module	350
VME 'DFG2' Framegrabber Card	2,270
Software License	0
Total	**9,320**

Table 4.3 *Hardware and Software Costing*

There are no software or maintenance costs. The cost of spares for this option would come to 9,320 euros.

Effort

Effort would be required for further research into Real-time Linux before the project could start. It is important to choose the right version of Real-time Linux. Effort would be used in creating the right development environment and learning new skills.

Estimated Time **9 months**

Risk

This option is given a medium risk. Although this option is feasible, it is still unsure on how it would be done, and indeed the best way to do the upgrade. The last thing the ING desire is for their version of Real-time Linux or their development tools to become obsolete or unsupported.

Risk Factor **Medium**

4.4. Option 2b Upgrade to Real-time Linux on x68 (cPCI)

This option would get the ING away from the OS-9 operating system but introduce a previously unused real-time operating system. A set of good open-source development tools may exist but wasn't found. This option gives more than a 30x increase in processor speed. It would time a relatively long time to finish this medium risk project.

Cost

Table 4.4 contains the prices (in euros) of the control hardware, software and software licenses that the ING would be required to purchase for this option.

Items	**Cost**
cPCI Crate	1,990
cPCI CPBI 1GHz Pentium 3	1,900
cPCI Transition Module	420
cPCI PMC Carrier Card	520
PMC 'Snapper' Framegrabber Card	2,610
cPCI IP Carrier Card	1,640
cPCI Motion Control Card	5,680
Software License	0
Total	**14,760**

Table 4.4 *Hardware and Software Costing*

There are no software costs or maintenance. The cost of spares for this option would come to 14,760 euros.

Effort

In addition to the effort required in option 2a, time would be need to find the best hardware and to learn more about the cPCI bus and I/O cards.

Estimated Time **9-12 months**

Risk

In addition to the risks documented for option 2a, there are risks in getting the new cPCI cards working together with the software.

Risk Factor **Medium**

4.5. Option 2c Upgrade to Real-time Linux on x68 (PCI)

This option would get the ING away from the OS-9 operating system but introduce a previously unused real-time operating system. A set of good open-source development tools may exist but wasn't found. This option gives more than a 30x increase in processor speed. It would time a relatively long time to finish this medium risk project.

Cost

Table 4.5 contains the prices (in euros) of the control hardware, software and software licenses that the ING would be required to purchase for this option.

Items	**Cost**
Pentium 3 Desktop	2,450
PCI Framegrabber Card	850
PCI Motion Controller Card	2,800
PCI Data Acquisition Card	940
Software License	0
Total	**7,040**

Table 4.5 *Hardware and Software Costing*

There are no software costs or maintenance. The cost of spares for this option would come to 7,040 euros.

Effort

In addition to the effort required in option 2a, effort would be required to exam the PCI backplane and to see if it could be used under the environmental conditions that come with operation in the observatory. Collaboration with the AAO may help to reduce time.

Estimated Time **9-12 months**

Risk

In addition to the risks documented for option 2a, there are risks in using a non-ruggedised PCI backplane. Collaboration with the AAO would help reduce risk.

Risk Factor **Medium**

4.6. Option 3a Upgrade to OS-9 on 68k (VME)

This option wouldn't get the ING away from the OS-9 operating system. It has a good supported set of development tools. It doesn't give us any gain in processing speed, using the same technology that was available 10 years ago. It would take a relatively short time to finish this low risk project.

Cost

Table 4.6 contains the prices (in euros) of the control hardware, software and software licenses that the ING would be required to purchase for this option.

Items	**Cost**
VME MVME147 Board	7,840
Software License	9,230
Total	**17,070**

Table 4.6 *Hardware and Software Costing*

The software and hardware costs are roughly equal. Software maintenance each year would come to 1,850 euros. The cost of spares for this option would come to 7,840 euros.

Effort

The option requires the least amount of effort. The main source of effort would be setting up the development environment and porting the code.

Estimated Time **3 months**

Risk

This option offers the lowest risk, all the original cards except the processor board would be used. With support from OS-9 in the porting of the code, the research has already shown that the upgrade is pretty much straightforward.

Risk Factor **Low**

4.7. Option 3b Upgrade to OS-9 on PowerPC (VME)

This option wouldn't get the ING away from the OS-9 operating system. It has a good supported set of development tools. It gives more than a 10x increase in processor speed. It would take a relatively short time to finish this low risk project.

Cost

Table 4.7 contains the prices (in euros) of the control hardware, software and software licenses that the ING would be required to purchase for this option.

Items	**Cost**
VME MVME5100 500MHz PowerPC	6,700
VME MVME761 Transistion Module	350
Software License	9,230
Total	**16,280**

Table 4.7 *Hardware and Software Costing*

The software and hardware costs are roughly equal. Software maintenance each year would come to 1,850 euros. The cost of spares for this option would come to 7,050 euros.

Effort

The option requires a small amount of effort of effort. The main source of effort would be setting up the development environment and porting the code.

Estimated Time **6 months**

Risk

There is low risk with this option, all the original cards except the processor board would be used. With support from OS-9 in the porting of the code, the research has already shown that the upgrade is pretty much straightforward.

Risk Factor **Low**

4.8. Summary

Table 4.8 below summarises the instrument cost (, required effort and risk associated with each option. Each figure is quoted in Euros and has been rounded to the nearest 10.

	Option						
	1a	**1b**	**2a**	**2b**	**2c**	**3a**	**3b**
Instrument Cost	21,010	38,210	9,320	14,760	7,040	17,070	16,280
Spares Cost	10,010	9,320	9,320	14,760	7,040	7,840	7,050
Maintenance Cost	8,340	9,760	0	0	0	1,850	1,850
Effort (months)	6	9	9	9-12	9-12	3	6
Risk Factor	Low	Low	Medium	Medium	Medium	Low	Low
Total Cost	**39,360**	**57,290**	**18,640**	**29,520**	**14,080**	**26,760**	**25,180**

Table 4.8 *Summary of Each options cost, effort and risk*

5. Recommendations

Again for reference, the seven options were;

??	Option 1a	Upgrade to VxWorks on 68k (VME)
??	Option 1b	Upgrade to VxWorks on PowerPC (VME)
??	Option 2a	Upgrade to Real-time Linux on PowerPC (VME)
??	Option 2b	Upgrade to Real-time Linux on x68 (cPCI)
??	Option 2c	Upgrade to Real-time Linux on x68 (Desktop - PCI)
??	Option 3a	Upgrade to OS-9 on 68k (VME)
??	Option 3b	Upgrade to OS-9 on PowerPC (VME)

It was seen from section 2.3 that having a standard operating system would increase efficiency. Development tools and applications would allow the software to be optimised and developed, while a faster processor would allow initial timesavings when performing an instrument set-up.

This instrument upgrade is crucial and needs to be done. It isn't desperate in terms of the instrument being operational, yet it is urgent, as it stops the instrument being developable, it currently simply can't be optimised. AF2 is one of the few instruments whose performance can improve with a software upgrade, as the upgrade makes further development possible. As the ING restructures, the staff available to work on software projects becomes less and crates a need for greater project and operational efficiency. An AF2 software upgrades offers more than other instrument upgrades and therefore is arguable more important. An upgrade will create a lot of opportunities for AF2, which is one of the ING flagship instruments.

This study has shown that if money was no object then Option 1b would be the most suitable upgrade option. It gives a standard operating system, excellent development tools and a 10x increase in processor speed, giving the instrument everything that is required to keep the instrument maintainable, upgradeable and developable until it's decommissioning. This would allow the ING project staff to optimise the instrument and enhance it, keeping it competitive.

The option is the most expensive, mainly because of software licensing, this includes the much needed development tools. Once the VxWorks PowerPC licenses are purchased, developing using another PowerPC for another instrument in the future would cost a lot less.

Even considering cost, it is the recommended solution. The cost difference may be over twice as much as the second recommended option, but the extra money is worth the better development tools and the development efficiency that comes using VxWorks.

It would be desirable to do this project in-house, by one of the members of the software group. Other options include contracting the work out or obtaining a postgraduate or student to work on it locally whilst being supervised.

A cheaper, and 2nd recommended solution is option 3b. It is chosen over 1a and 3a because of the processor. AF2 will be around for another 10 years yet there are signs that the 68k processor won't be supported in 10 years from now. It is already over 10 years old and VxWorks AE and Real-time Linux currently don't support the processor. If an upgrade is going to last until AF2 is decommissioned then the PowerPC is a better bet.

The disadvantage of this option is that it would still be on OS-9, although a lot of the development inefficiency would go because of the set of good development tools. It is also noted that option 3b costs less than 1a and 3a.

Upgrading the software to VxWorks but using a 68k processor, option 1a, is still an improvement as it allows the software to be developed with an excellent set of real-time tools. The code can be optimised saving some time on configuring a field. Using a 68k processor on OS-9, option 3a, would still be an improvement on what the ING currently have, the project would be short, making development tools available. Therefore 1a and 3a are the 3rd and 4th choices respectively.

It is felt that, at this stage, Real-time Linux could not be recommended. All options are feasible but it is felt that the software is better left alone at this time rather than to upgrade to a version of Real-time Linux.

Real-time Linux upgrades were investigated for the same reasons it is being used or considered elsewhere, because it is cheap and requires no software licenses. Three options were investigated because it wasn't known which option was going to be the cheapest. As it is possible to re-use some of the VME cards, option 2a was found to be just a little more expensive than using a desktop with PCI cards, option 2c. Option 2b, the cPCI option, was found to cost more, take longer and offer no benefits over the other Linux and the OS-9 options, therefore it should not be a considered.

Real-time Linux isn't recommended mainly because of the lack of research and understanding of the operating system. It is still evolving and it would be wise for the ING to increase their knowledge and understanding of this Real-time operating system. Recommending Real-time Linux at this stage would be a mistake.

It is recommended that a PC is set-up and Real-time Linux is installed on it, so that it can be used as an experimental machine. It is recommended that the ING pursue Linux because it will become more and more popular within the world of dedicated instrumentation. Learning from 6df and the AAO would be a good starting place.

If, for some reason Real-time Linux was decided upon then option 2a would be preferred over 2c, mainly because of the length of time taken and the uncertainties, as there are less new things to consider and research.

In conclusion, giving AF2 a solid environment is crucial. Upgrading the software to VxWorks running on a PowerPC was found to be the best solution. It would make the software maintainable, developable and upgradeable. Allowing the instrument to remain competitive.

6. References

The following documents are relevant:

[1] "*Case for AF2 Software Upgrade Feasibility Study*", S. Goodsell, ING
[2] "*http://csgsrv.ing.iac.es/af2/*", S. Goodsell, ING
[3] "*http://csgsrv.ing.iac.es/af2/af2_documentationset.html/*", S. Goodsell, ING
[4] "*AUTOFIB-2/WYFFOS User Manual*", J. Telting, R. Corradi, ING
[5] "*AUTOFIB-2 Technical Manual*", I. J. Lewis, University of Durham
[6] "*TP32V User Manual*", Tadpole Technology Plc
[7] "*BVME780 Frame Grabber and Graphics Module*", BVME
[8] "*DMC-300-10 DC Motor Controller User's Manual*", DMC
[9] "*VIPC310 User Manual*", Greenspring Computers
[10] "*IP-Digital 24 User Manual*", Greenspring Computers
[11] "*IP-Precision ADC*", Greenspring Computers
[12] " AUTOFIB-2 Control System Software Manual", I. J. Lewis, University of Durham
[13] "Visiting Committee 2001 Report", [FIND OUT]
[14] "ING Restructure Plan", R Rutten, ING [Check]
[15] " AF2 Operation's Analysis", S. Goodsell, ING
[16] "*http://www.embedded.com/97/fe39710.htm*", J. Epplin, Embedded.com
[17] "*Introduction To Real-Time Systems Programming*", RTP Limited
[18] "*http://www.microware.com*", Microware Systems
[19] "*http://www.radisys.com/*", RadiSys Corporation
[20] "*OS-9 for 68k Datasheet*", RadiSys Corporation
[21] "*OS-9 for PowerPC Datasheet*", RadiSys Corporation
[22] "*OS-9 Technical Manual v3.0*", Microware Systems
[23] "*Hawk IDE Datasheet*", RadiSys Corporation
[24] Personal email correspondence, Jean-Pierre Van Renterghem, Microware
[25] "*http://www.microware.com/Support/levels.html*", Microware Systems
[26] "*http://www.windriver.com/*", Wind River Systems Inc.
[27] "*VxWorks 5.x Datasheet*", Wind River Systems Inc.
[28] "*VxWorks AE Datasheet*", Wind River Systems Inc.
[29] "*VxWorks Programmer's Guide*", Wind River System Inc.
[30] "*Tornado II Datasheet*", Wind River Systems Inc.
[31] "*Tornado II Whitepaper*", Wind River Systems Inc.
[32] "*Tornado User's Guide v2.0*", Wind River System Inc.
[33] "*WindView User's Guide v2.0.1*", Wind River System Inc.
[34] "*StethoScope User's Manual v5.3*", Wind River System Inc.
[35] "*TraceScope User's Manual v1.0*", Wind River System Inc.
[36] "*ProfileScope User's Maunual v3.0*", Wind River System Inc.
[37] "*Customer Support User Guide v5.0*", Wind River System Inc.
[38] "*http://www.linuxdevices.com/articles/AT8073314981.html*", LinuxDevices.com
[39] "*http://www.uk.research.att.com/~dmi/linux-srt/*", D Ingram, AT&T
[40] "*http://www.timesys.com/*", TimeSys
[41] "*http://www.lineo.com/products/embedix_sdk/embedix/index.html*", Lineo
[42] "*http://www.lynuxworks.com/products/bluecat/bluecat.php3*", LynuxWorks
[43] "*http://www.linuxdevices.com/links/LK8662675028.html*", LinuxDevices.com
[44] "*J1 + J2 Monolithic Backplane Datasheet*", Bustronic Ltd
[45] "*CPCI Backplane with ATX Datasheet*", Bustronic Ltd

[46] “*http://www.aao.gov.au/ukst/6df.html*”, Q. Parker, AAO
[47] “*6dF Positioner Task Software Report*”, J Wilcox, T Farrell & N Frampton, AAO
[48] Personal email correspondence, Janet Eden, Tadpole Technology Ltd
[49] “*MVME162P2 VME Embedded Controller Datasheet*”, Motorola PLC
[50] “*MVME172P2 VME Embedded Controller Datasheet*”, Motorola PLC
[51] “*MVME147 VME Single Board Computer Datasheet*”, Motorola PLC
[52] “*MVME167P VME Single Board Computer Datasheet*”, Motorola PLC
[53] “*MVME177P VME Single Board Computer Datasheet*”, Motorola PLC
[54] “*MVME2100 Series VME Processor Modules Datasheet*”, Motorola PLC
[55] “*MVME2300 Series VME Processor Modules Datasheet*”, Motorola PLC
[56] “*MVME2400 Series VME Processor Modules Datasheet*”, Motorola PLC
[57] “*MVME2600 Series VME Processor Modules Datasheet*”, Motorola PLC
[58] “*MVME2700 Series VME Processor Modules Datasheet*”, Motorola PLC
[59] “*MVME3400 & MVMVE4600 VME Processor Modules Datasheet*”, Motorola PLC
[60] “*MVME5100 Series VME Processor Modules Datasheet*”, Motorola PLC
[61] “*CPN5360 CompactPCI Peripheral Processor Datasheet*”, Motorola PLC
[62] “*CPLE cPCI SBC Datasheet*”, Trenton Technology Inc
[63] “*CPBI cPCI SBC Datasheet*”, Trenton Technology Inc
[64] Personal email correspondence, Jason Young, Arcom Control Systems
[65] Personal email correspondence, Stuart Tier, BVM Ltd
[66] “*Snapper-8 Datasheet*”, Datacell Ltd
[67] “DT3155 Framegrabber PCI Card Datasheet”, Data Translation Inc
[68] “*Galil Motion Controller Optima Series Data Pages*”, Galil Motion Control Inc.
[69] Personal email correspondence, Kaushal Shah, Galil Motion Control Inc
[70] “*http://www.dynatem.com/cipcfs.html*”, Dynatem Inc.
[71] “*IP-Unidig Datasheet*”, SBS Technologies Inc.
[72] “*DT302 Data Acquisition Card Datasheet*”, Data Translation Inc
[73] “*IP-OPTOAD12 Datasheet*”, SBS Technologies Inc.

At the time of releasing this document all the references, with the exception of web links, were available in the author’s office.

Appendix A Software Tasks

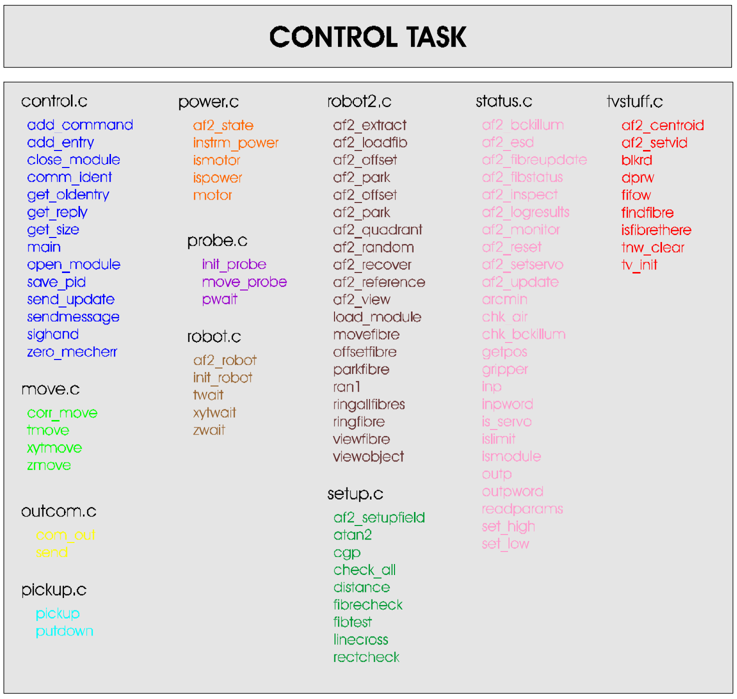
This Appendix looks at the functions associatated with each of the AF2 tasks running on the OS-9 crate. It also lists the function calls for each of the tasks functions. This appendix is best viewed in colour.

Table A1 lists the 7 tasks as well as the addition 2 OS-9 specific tasks running on the crate, the table is a copy of table 2.3.

Task	Type	Description
Control	Control	Main Control Task
Cmdintrp	Control	Command Interpreter
Read	Network	Network Layer Read Task
Receive	Network	Network Layer Receive Task
Write	Network	Network Layer Write Task
Watchdog	Network	Network Layer Watchdog Task
Esdrave	Engineering	Engineering Mimic
Shell	OS-9	Command Line Shell
Procs	OS-9	Procs process displaying the information

Table A1 *AF2 OS-9 Software Tasks*

Control Task



CONTROL TASK

control.c
add_command
add_entry
close_module
comm_ident
get_oldentry
get_reply
get_size
main
open_module
save_pid
send_update
sendmessage
sighand
zero_mecherr

move.c
corr_move
tmove
xytmove
zmove

outcom.c
com_out
send

pickup.c
pickup
putdown

power.c
af2_state
instrm_power
ismotor
ispower
motor

probe.c
init_probe
move_probe
pwait

robot.c
af2_robot
init_robot
twait
xytwait
zwait

robot2.c
af2_extract
af2_loadfib
af2_offset
af2_park
af2_offset
af2_park
af2_quadrant
af2_random
af2_recover
af2_reference
af2_view
load_module
movefibre
offsetfibre
parkfibre
ran1
ringallfibres
ringfibre
viewfibre
viewobject

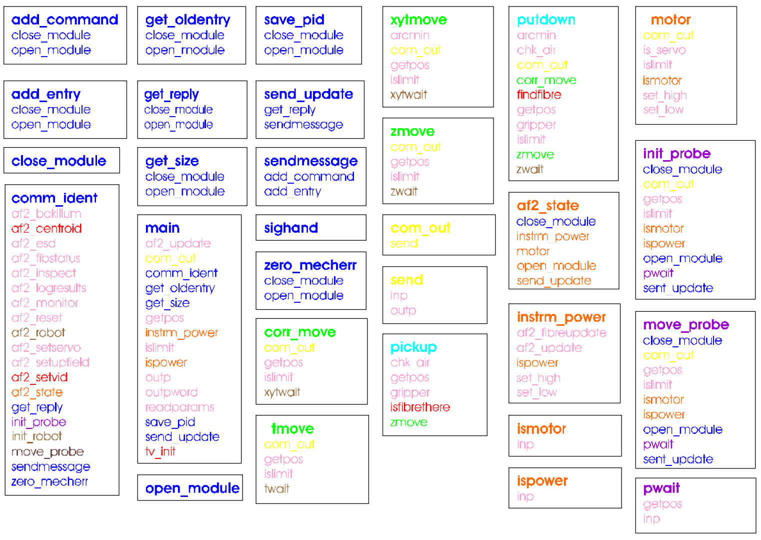
setup.c
af2_setupfield
atan2
cgp
check_all
distance
fibrecheck
fibtest
linecross
rectcheck

status.c
af2_bckillum
af2_esd
af2_fibreupdate
af2_fibstatus
af2_inspect
af2_logresults
af2_monitor
af2_reset
af2_setservo
af2_update
arcmin
chk_air
chk_bckillum
getpos
gripper
inp
inpword
is_servo
islimit
ismodule
outp
outpword
readparams
set_high
set_low

tvstuff.c
af2_centroid
af2_setvid
blkrd
dprw
fifow
findfibre
isfibrethere
tnw_clear
tv_init

Figure A1 *Function's contained within the Control Task*

The control task is by far the biggest task. It consists of 12 source code files containing 102 AF2 functions as can be seen in Figure A1. Every AF2 function was examined to discover which functions are being called. Figures A2, A3 and A4 contain a list of every function with it's associated function calls. The colour of the function indicates which file it is defined in, this is visual from figure A1. The non-AF2 functions are in black.



add_command
close_module
open_module

add_entry
close_module
open_module

close_module

comm_ident
af2_bckillum
af2_centroid
af2_esd
af2_fibstatus
af2_inspect
af2_logresults
af2_monitor
af2_reset
af2_robot
af2_setservo
af2_setupfield
af2_setvid
af2_state
get_reply
init_probe
init_robot
move_probe
sendmessage
zero_mecherr

get_oldentry
close_module
open_module

get_reply
close_module
open_module

get_size
close_module
open_module

main
af2_update
com_out
comm_ident
get_oldentry
get_size
getpos
instrm_power
islimit
ispower
outp
outpword
readparams
save_pid
send_update
tv_init

open_module

save_pid
close_module
open_module

send_update
get_reply
sendmessage

sendmessage
add_command
add_entry

sighand

zero_mecherr
close_module
open_module

corr_move
com_out
getpos
islimit
xytwait

tmove
com_out
getpos
islimit
twait

xytmove
arcmin
com_out
getpos
islimit
xytwait

zmove
com_out
getpos
islimit
zwait

com_out
send

send
inp
outp

pickup
chk_air
getpos
gripper
isfibrethere
zmove

putdown
arcmin
chk_air
com_out
corr_move
findfibre
getpos
gripper
islimit
zmove
zwait

af2_state
close_module
instrm_power
motor
open_module
send_update

instrm_power
af2_fibreupdate
af2_update
ispower
set_high
set_low

ismotor
inp

ispower
inp

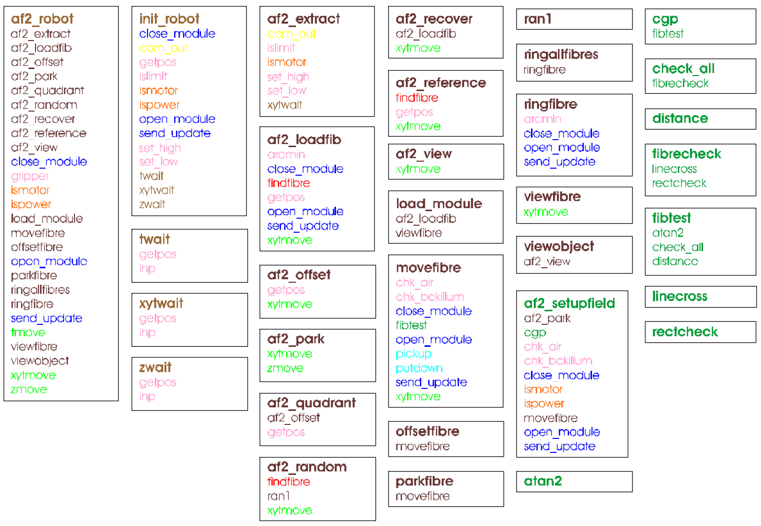
motor
com_out
is_servo
islimit
ismotor
set_high
set_low

init_probe
close_module
com_out
getpos
islimit
ismotor
ispower
open_module
pwait
sent_update

move_probe
close_module
com_out
getpos
islimit
ismotor
ispower
open_module
pwait
sent_update

pwait
getpos
inp

Figure A2 *Function Calls*



af2_robot
af2_extract
af2_loadfib
af2_offset
af2_park
af2_quadrant
af2_random
af2_recover
af2_reference
af2_view
close_module
gripper
ismotor
ispower
load_module
movefibre
offsetfibre
open_module
parkfibre
ringallfibres
ringfibre
send_update
tmove
viewfibre
viewobject
xytmove
zmove

init_robot
close_module
com_out
getpos
islimit
ismotor
ispower
open_module
send_update
set_high
set_low
twait
xytwait
zwait

twait
getpos
inp

xytwait
getpos
inp

zwait
getpos
inp

af2_extract
com_out
islimit
ismotor
set_high
set_low
xytwait

af2_loadfib
arcmin
close_module
findfibre
getpos
open_module
send_update
xytmove

af2_offset
getpos
xytmove

af2_park
xytmove
zmove

af2_quadrant
af2_offset
getpos

af2_random
findfibre
ran1
xytmove

af2_recover
af2_loadfib
xytmove

af2_reference
findfibre
getpos
xytmove

af2_view
xytmove

load_module
af2_loadfib
viewfibre

movefibre
chk_air
chk_bckillum
close_module
fibtest
open_module
pickup
putdown
send_update
xytmove

offsetfibre
movefibre

parkfibre
movefibre

ran1

ringallfibres
ringfibre

ringfibre
arcmin
close_module
open_module
send_update

viewfibre
xytmove

viewobject
af2_view

af2_setupfield
af2_park
cgp
chk_air
chk_bckillum
close_module
ismotor
ispower
movefibre
open_module
send_update

atan2

cgp
fibtest

check_all
fibrecheck

distance

fibrecheck
linecross
rectcheck

fibtest
atan2
check_all
distance

linecross

rectcheck

Figure A3 *Function Calls*

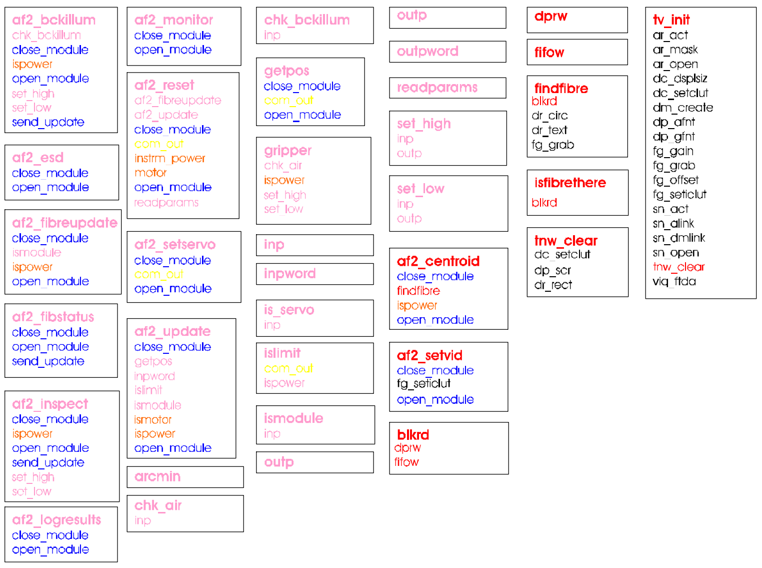
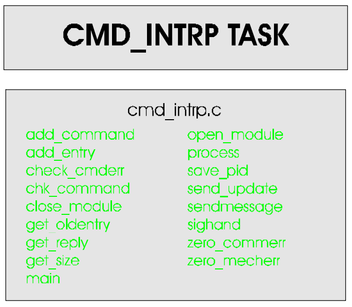


Figure A4 *Function Calls*

13 of the 102 AF2 functions, call a total of 25 non-AF2 functions. 7 of these functions are associated with data modules, 16 are associated with the graphics card and the other 2 are associated with the OS-9 system.

The other 89 functions contained may contain calls to these functions but providing the function interface remains the same, the other 89 functions needn't change.

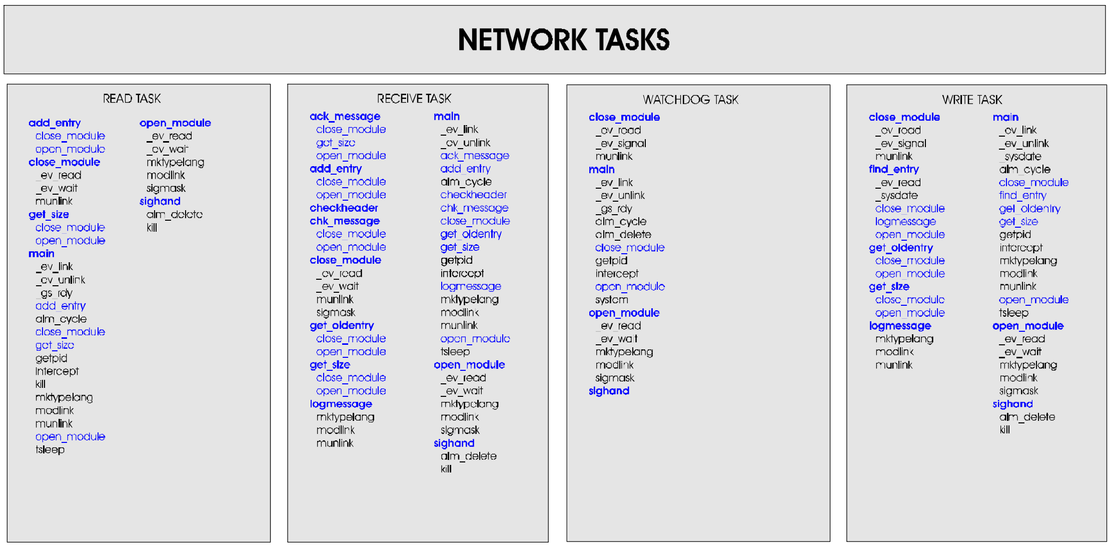
Cmd_intrp Task



It contains 17 functions. 5 of these function calls non AF2-functions. 3 of these are duplicates, of which 12 are very similar/duplicates from the control task, although a couple have slight modifications.

From the other 5, 2 call non-AF2 functions. 1 of these calls a system function that isn't called in the control task.

Network Tasks

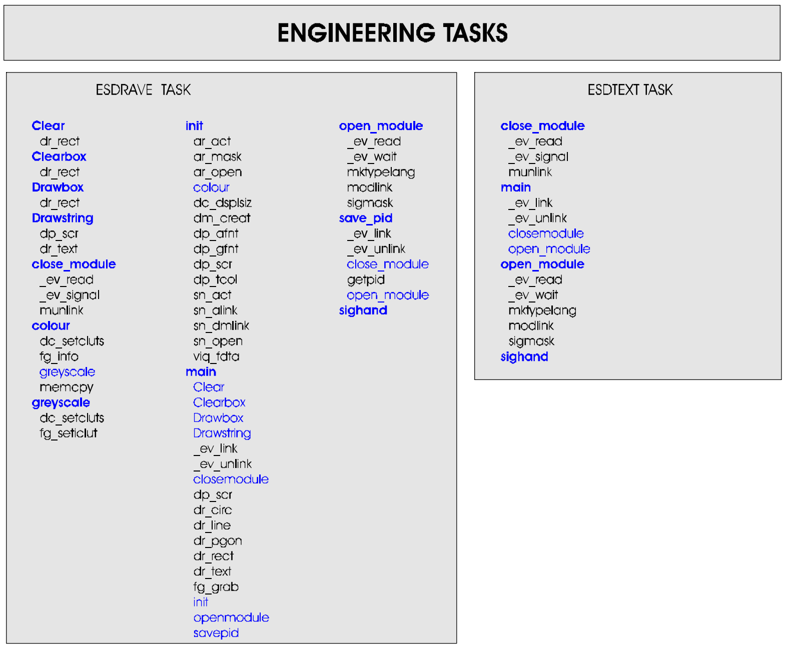


There are 4 .c files in the Network directory. The read.c (5/6), receive.c (8/11), watchdog.c (3/4) and write.c (6/8) create the read, receive, watchdog and write tasks collectively. Altogether there are 29 functions with each function being slightly different. Some of these functions use system calls specific to alarms and datamodules. There are 3 different non-AF2 functions called.

Out of the 29 functions, 18 contain non-AF2 functions. Out of these 9 are very similar.

Engineering Tasks

The files esdrave.c and esdtext.c form the engineering tasks. They consist of 16 functions all which contain non-AF2 function calls for mainly the graphics/Framegrabber cards, also a few datamodule calls.



Other Software

Although not tasks the table.c file, that crates the datamodules and the eng.c tasks that crates the engineering command line should also be considered.

The file table.c just consists of a main function and uses function calls to set up the datamodules. The eng.c contains 11 functions, 3 contain system calls, 2 are repeats.

Summary

In summary there are 173 functions. Out of these functions 53 contain system calls. Out of these 53 , as 13 of these functions are duplicates, this leaves us with 40 functions. This roughly makes the software 75% portable.

Appendix B VxWorks Licensing

The licensing is complicated. There are two main parts to the VxWorks license, the development license and the run-time license.

The ING currently have a Tornado Professional License for the 68K microprocessor architecture family, a VxWorks OEM development license for the 68020 processor and a board support package for a Board support package for the MVME147. The ING, however don’t have any run-time licenses.

B.1 Development Licenses

There are three parts to the development licensing. These are the Tornado Professional Licence, the VxWorks OEM Development License and the Board Support Package.

Tornado Professional License

This is a license for the development tools. It is licensed to a specific company, at a site, for a specific host computer, a target microprocessor architecture family and a specified number of users.

If the ING doesn’t change the microprocessor architecture family (i.e. Solaris cross 68K) then they will not have to purchase new ones.

If the PowerPC option is chosen then the ING will have to buy a new one costing 8,340 euros with annual maintenance being 1,420 euros. This price is per user.

For reference, the 68K (microprocessor architecture family) includes the 68000, 68010, 68020, 68040 and 68060 processors. The PowerPC includes the MPC7xx, MPC74xx, MPC6xx, MPC8xx, MPC82xx, MPC5xx and PPC4xx processors.

VxWorks OEM development License

This is the kernel, compiled for a given processor family (i.e. 68020). It is licensed to a specific company for a project (customer product, site and processor family). If the ING would start a new project then they would have to buy one. The project the ING are registered for is named ‘Telescope’ which would cover all instrumentation projects.

The current OEM license the ING have is the Solaris cross 68020, if the ING started to work with the 68060, in the same project, they would get a 50% discount costing 9.630 euros with 8.340 euros annual maintenance, because it is from the same family.

If the ING with to purchase a OEM license for the MPC7xx processor, they would have to pay 19.260 euros, with 8.340 euros annual maintenance. With MPC7xx including the MPC750, MPC755 and MPC740 processors.

Board Support Package (BSP)

The BSP is for a given board, for example, the MVME147, the MVME2100,… etc.

Each new OEM development license includes one. If a board if used from the same processor family (68020), then the ING would have to buy a new BSP, costing 1.930 euros with 330 euros annual maintenance.

So for example, if the ING was to start developing with the 68060 processor, one BSP will be included in the new VxWorks OEM License. This is equally valid for the MPC7xx processor.

B.2 Run-time licenses

According to the number of copies of VxWorks you do at the production time. The ING would have to purchase one run-time license, costing 1,290 euros.

Appendix C Experiments

Experiments were performed to estimate the affects of using a faster processor. They also gave a rough idea of the sort of timesavings that could be gained by implementing changes to the code. The experiments tried to measure and allocate time to either moving mechanisms or data processing. Data processing includes processes that would use the processor, file I/O and the movement of data.

The experiments are repeatable. Each experiment started with the robot in the park position and the fibres in a 'park' configuration. There were a total of 159 fibres used. The experiments were timed using a stopwatch, as initial experiments using an OS-9 system timer call failed.

The first experiment that was carried out was to look to see how long it takes to move the robot to each fibre in sequential order before returning to the park position. The 'Viewmodule' routine is mainly movement.

The second experiment was to record the position of all the fibres. The robot would move to the first fibre from the park position, and perform a centroid. The robot moved to and performed a centroid on each fibre before returning to the park position. This routine is known as 'Loadmodule'. The difference between the two routines is purely due to data processing.

Table C1 shows the results of these experiments. The total time centroiding, and the centroiding time for each fibre is deduced.

	Total Time (sec)	Time per Fibre (sec)
viewfibre	183	1.15
Loadmodule	380	2.39
Centroiding	**197**	**1.24**

Table C1 *Deducing Timing Of Centroiding*

The third experiment was to see how long to took to move from one configuration field to another, without placing the fibres to any accuracy. The fibres went from a 'park' configuration to a 'circle' configuration.

This means that the robot moves to the first fibre, performs a centroid, picks up the fibre, moves to the new position, places the fibre, and then centroids again to record where it is. The robot then moves to the second fibre and repeats the process for all fibres. Before the robot moves the software checks a configure algorithm to make sure there will be no collisions on setting the field up. Table C2 displays the results, the pick-move-place time for each fibre is deduced from this experiment.

	Total Time (sec)	Time per Fibre (sec)
Park	1718	
Configure Algorithm	65	
Loadmodule	380	2.39
Centroid	197	1.24
Pick-Move-Place	**1076**	**6.77**

Table C2 *Changing field without iterating*

The final experiment was again to move the fibres from the ‘park’ configuration to the ‘circle’ configuration. This time the fibres were to be placed within a 8 micron placement accuracy. If the fibre was placed outside this tolerance then it would be re-iterated. In this experiment there was a total of 309 fibre centroids. This means that there was an extra 150 movements where the robot picked up and slightly moved before trying to place the fibre within tolerance again.

Table C3 contains the results and deduces the total reiteration movement time and the time spent of each reiteration.

	Time per Fibre (sec)	Total Time (sec)
Field Set-up		2531
Configuring		65
Viewmodule	1.15	183
Centroid	1.24	197
Centroid (x309)	1.24	383
Pick-Move-Place	6.77	1076
Reiterate Move (x150)	**4.18**	**627**

Table C3 *Changing field with iterating*

These experiments have tried to break up the different commands into parts to estimate how much time could be saved. It is clear that the centroid algorithms and the configuration algorithm are purely data processing parts of the software. If there was a 10x increase in processor speed then it is expected there would be a 90% time saving for these algorithms. This alone would lead to a 20% time saving on a field configuration.

Further small timesaving would relate to the Pick-Move-Place and Reiterate Move parts of the process.

It is also interesting to note that if a lookup table containing each fibre/fin misalignment was introduced, it would reduced the time by about 30%. This is the total time taken between the third and final experiment.

Finally if a faster processor was used and a lookup table introduced then the total saving would be over 40%.

Appendix D

This Appendix contains all the prices (in euros) of the hardware, software and licenses obtained as part of the research.

Item	TYPE	Description	Manufacture	Price
CPCI1990000196	Backplane	A 7U 19" Chassis with a 6 slot CPCI backplane	Bustronics	1,990
PC + PCI Backplane	PC + Backplane	Industril PC (x86) P3, 1 GHz, 64 Mb DRAM	Arcom Control Systems	2,450
MVME147-024A	VME	Single-Board Computer 68k, 33 MHz, 32 MB, DRAM	Motorola	7,840
MVME167PA-36SE	VME	Single-Board Computer 68k, 33 MHz, 64 Mb SDRAM	Motorola	12,160
MVME5110-2263	VME	Processor Module (PowerPC) MPC7410, 512 Mb ECC SDRAM	Motorola	6,700
MVME761-001	VME	I/O Transition Module	Motorola	350
CPBI-5814-110-0M	cPCI	Single Board Computer (x86) P3 1 GHz	Trenton Technology	1,920
RTM10-5819-000	cPCI	I/O Transition Module	Trenton Technology	420
CPN5360B-500-08	cPCI	Peripheral Processor (x86) P3, 500 MHz, 256 Mb SDRAM	Motorola	3,880
CPTM-01	cPCI	I/O Transition Module	Motorola	420
DFG2	VME	Framegrabber Card	Dynatem	2,270
Snapper8 + SDK	PMC	Framegrabber	Datacell	4,240
DT3155	VME	Framegrabber	Data Translation	1,140
DMC-1348	VME	Motion Control Card (4 axis)	Galil	2,840
DMC-1680	CPCI	Motion Control Card (4 axis)	Galil	2,840
DMC-1842	PCI	Motion Control Card (8 axis)	Galil	3,640
VIPC310	VME	IP Carrier Card	SBS Technologies	350
CIPC	CPCI	IP Carrier Card	Dynatem	1,640
IP-Unidig	IP	IP DI/O Card	SBS Technologies	410
DT302	PCI	Data Acquisition Card	Data Translation	940
IP-OPTOAD12	IP	IP ADC Card	SBS Technologies	1,040
CPCI	cPCI	PMC Carrier Card	Dynatem	520
Snapper8	PMC	Framegrabber	Datacell	2,610

Table D1 *Prices of Control Hardware*

Table D1 contains the prices on the control hardware described in section 3.2 and used in section 4. Table D2 describes the prices of the different software licenses and maintenance. All prices have been rounded up to the nearest £10.

Option	VxWorks Tornado	VxWorks OEM	VxWorks BSP	VxWorks Run-time	OS-9 License	License Price	Annual Maintenance Price
1a	0	9,630	0	1,290	N/A	10,920	8,340
1b	8,340	19,260	0	1,290	N/A	28,890	9,760
2a	N/A	N/A	N/A	N/A	N/A	0	0
2b	N/A	N/A	N/A	N/A	N/A	0	0
2c	N/A	N/A	N/A	N/A	N/A	0	0
3a	N/A	N/A	N/A	N/A	9,230	9,230	1,850
3b	N/A	N/A	N/A	N/A	9,230	9,230	1,845

Table D2 *Software Licensing and Maintenance Prices*