MEMORIA TFC

"Análisis del proceso de migración de aplicación embebida en tiempo real para estación del segmento de tierra del sistema de aumentación satélite europeo"

Alumno: Samuel Fernández León

Contenidos

1. INTRODUCCIÓN Y VISTA GENERAL DEL DOCUMENTO 5
1.1 BIBLIOGRAFÍA Y FUENTES DE INFORMACIÓN 5

2. DEFINICIÓN Y MARCO Y OBJETIVOS DEL TFC 7
2.1 EL SISTEMA EGNOS 7
2.2 LAS ESTACIONES RIMS 7
2.3 EL CORE COMPUTER DE LA RIMS 10
2.4 OBJETIVOS DEL TFC 11
2.5 LISTA DE TAREAS 12
2.5.1 Diseño HW 12
2.5.2 Adquisición de componentes 12
2.5.3 DIseño detallado 12
2.5.4 Migración de la BSP de acuerdo a las necesidades del CC SW 12
2.5.5 Adaptación del SW & AIV (Assembly, Integration & Validation) 13
2.5.6 Marcado CE 13
2.5.7 Tests de Non-Regression 13
2.5.8 Actividades de cualificación de los COTS 13
2.6 PLANIFICACIÓN 14

3. DISEÑO DETALLADO 15
3.1 ANÁLISIS HW/SW 15
3.1.1 Arquitectura hardware 15
3.1.2 Análisis del sistema operativo (LynxOS 3.1.0a) 19
3.1.2.1 Funcionalidades del sistema operativo 19
3.1.2.2 Advanced Porting Kit 21
3.1.2.3 Estructura de la BSP 22
3.1.2.4 Drivers de dispositivos en LynxOS 22
3.1.2.4.1 Estructura de los drivers de LynxOS 22
3.1.2.4.2 Funciones de entrada (entry point functions) 23
3.1.2.4.3 Estructura de datos de dispositivo 24
3.1.2.4.4 Estructura de datos estática 24
3.1.2.4.5 Gestión de memoria 24
3.1.2.4.6 Sincronización 25
3.1.2.5 Manejo de interrupción y del timeout 25
3.1.3 FUNDAMENTOS DE DISEÑO DE LA BSP DE LA PLACA VMPC6a 26
3.1.3.1 Visión general de la BSP de VMPC6a 26
3.1.3.2 Estructura de la BSP para VMPC6a 26
3.1.3.3 Controladores de dispositivos de la BSP para VMPC6a 27
3.1.4 NECESIDADES DE SOPORTE HARDWARE DE LA BSP PARA VM6250 27
3.1.4.1 REUSABILIDAD 29
3.1.4.2 PROCESO DE ARRANQUE DE LA PLACA VM6250 30
3.1.4.3 DRIVERS DE DISPOSITIVOS PARA VM6250 30
3.1.4.3.1 Puertos serie (TTY) 31
3.1.4.3.2 CPLD (Watchdog y GPIO) 31
3.1.4.3.3 Interfaz de red 31
3.1.4.3.4 Driver SATA 31
3.1.4.3.5 Driver de la placa TEWS TPMC866-11R 31
3.2 INTEGRACIÓN HARDWARE DEL CORE COMPUTER 32
3.2.1 Integración hardware del Core Computer 32
3.2.2 Modificaciones hardware 36
3.2.2.1 Gestión de LEDs 36

3.2.2.2 Puertos serie 37
4. MIGRACIÓN DE LA BSP 39
5. ADAPTACIÓN DEL SW & AIV 41
5.1 PROCESO DE MIGRACIÓN DE LA APLICACIÓN 41
5.2 ESTRATEGIA DE TEST 42
5.2.1 Validación de la BSP 43
5.2.2 Unit testing 44
5.2.3 Validación de la aplicación a nivel de CC 45
5.2.4 Validación de la aplicación a nivel de estación 47
6. MARCADO CE 49
6.1 NORMATIVA APLICABLE 49
6.2 ESTRATEGIA DE CERTIFICACIÓN 50
7. ESTRATEGIA DE CUALIFICACIÓN 51
8. TAREAS DESARROLLADAS DURANTE LA EJECUCIÓN DEL PROYECTO 53
9. CONCLUSIONES 54

Lista de Figuras

FIGURA 1 – COMPOSICIÓN DEL SISTEMA EGNOS 8
FIGURA 2 – DISPOSICIÓN DE LAS ESTACIONES DEL SEGMENTO DE TIERRA DE EGNOS 8
FIGURA 3 - DIAGRAMA DE BLOQUES DE LA ESTACIÓN RIMS 10
FIGURA 4 – PLANIFICACIÓN DEL PROYECTO 14
FIGURA 5- ARQUITECTURA DE LA PLACA VMPC6A 18
FIGURA 6 - ARQUITECTURA DE LA PLACA VM6250 19
FIGURA 7 - ARQUITECTURA SOFTWARE 21
FIGURA 8 - ESTRUCTURA DE LA BSP 22
FIGURA 9 - ESTRUCTURA DE LOS DRIVERS DE LYNXOS 23
FIGURA 10 - LEDS DE ESTADO DEL NUEVO CHASSIS (DERECHA) 36
FIGURA 11 - ESQUEMA ELÉCTRICO DE LOS LEDS DE ESTADO 37
FIGURA 12 - PUERTOS SERIE ADICIONALES 38
FIGURA 13 - ESTRATEGIA DE TEST 42
FIGURA 14 - ENTORNO DE VALIDACIÓN DEL CORE COMPUTER 46
FIGURA 15 - ENTORNO DE VALIDACIÓN DE LA ESTACIÓN 48

Lista de Tablas

TABLA 1 - DIFERENCIAS ENTRE LAS PLATAFORMAS VMPC6A Y VM6250 16
TABLA 2 - ENTRY POINT FUNCTIONS 24
TABLA 3 - COMPONENTES DE LA BSP PARA VMPC6A 26

TABLA 4 - NECESIDADES DE SOPORTE HW PARA VM6250 27
TABLA 5 - INTERFACES A IMPLEMENTAR 28
TABLA 6 - PARTES DEL SO A MODIFICAR 30
TABLA 7 - DRIVERS A IMPLEMENTAR PARA VM6250 31
TABLA 8 – PRINCIPALES COMPONENTES HARDWARE DEL CORE COMPUTER 32
TABLA 9 – COMPARATIVA DE INTERFACES 34
TABLA 10 – TESTS DE VALIDACIÓN DE LA BSP 44

1. INTRODUCCIÓN Y VISTA GENERAL DEL DOCUMENTO

El presente documento contiene la memoria del trabajo fin de carrera titulado “Análisis del proceso de migración de aplicación en tiempo real para estación del segmento de tierra del sistema EGNOS”.

Este proyecto basado en mi actividad laboral como ingeniero de desarrollo para una empresa que realiza entre otras muchas cosas, proyectos de navegación por satélite para la Agencia Espacial Europea (ESA).

El trabajo describe el proceso de migración de una aplicación embebida en tiempo real a una nueva plataforma hardware enmarcada el proyecto EGNOS de la ESA.

La memoria consta de los siguientes contenidos:

- Capítulo 1: contiene la introducción del documento y las referencias bibliográficas
- Capítulo 2: explica los objetivos del TFC y la necesidad de la que surge
- Capítulo 3: detalla el análisis HW/SW realizado y el proceso de diseño de la solución final
- Capítulo 4: explica el proceso de adaptación de la BSP
- Capítulo 5: explica el proceso de migración de la aplicación, integración y testeado del sistema
- Capítulo 6: describe las actividades realizadas relacionadas con el marcado CE del hardware
- Capítulo 7: introduce la problemática relacionada con la cualificación de SW crítico y detalla la estrategia a seguir para cualificar el sistema migrado
- Capítulo 8: detalla las tareas que el autor del presente TFC ha realizado en la empresa en el marco de este proyecto
- Capítulo 9: contiene unas conclusiones y reflexiones finales sobre el proyecto

1.1 BIBLIOGRAFÍA Y FUENTES DE INFORMACIÓN

- Sitio web de la Agencia Espacial Europea
  www.esa.int
- Sitio web de LynuxWorks
  www.lynuxworks.com
- Sitio web de Kontron
  www.kontron.com

- Manual de la placa TEWS TPMC866-11R
  Ref. D76866801 v1.0

- Manuales de la placa VMPC6a

VMPC6a User's Guide	Ref. CA.DT-129-5e
VMPC6a HW release notes	Ref. CA.DT.328-6e
VMPCBug User's Manual	Ref. SD.DT.A35-Le

- Manuales de la placa VM6250

VM6250 User's Guide	Ref. CA.DT.A65-0e
U-boot User's Manual	Ref. SD.DT.F36-0e
PBIT User's Guide	Ref. SD.DT.F35-0e

- Documentación de la BSP (parte del porting kit)

- Documentación entregada a ESA (development plan, detailed design, design justification…). Estos documentos son confidenciales.

- Documentación preparada por la empresa durante el desarrollo del la antigua estación . Estos documentos son confidenciales

2. DEFINICIÓN Y MARCO Y OBJETIVOS DEL TFC

2.1 EL SISTEMA EGNOS

El sistema EGNOS (European Geostationary Navigation Overlay Service) es el primer sistema de navegación europeo. Es capaz de aumentar la precisión del sistema de navegación GPS y lo hace utilizable para aplicaciones críticas como guiar maniobras de aproximación de aviones o la navegación de barcos a través de canales estrechos.

Consiste en 3 satélites geoestacionarios y una red de estaciones de tierra. EGNOS alcanza sus objetivos transmitiendo una señal que contiene información sobre la fiabilidad y la precisión de las señales de posicionamiento enviadas por GPS y permite a los usuarios en Europa determinar su posición con un error de 1.5 metros.

EGNOS es un proyecto conjunto de la ESA, la Comisión Europea y Eurocontrol, la organización europea para seguridad en navegación aérea. Se trata de la primera actividad europea en el marco de los sistemas globales de navegación por satélite (GNSS) y el precursor de Galileo, el sistema de navegación por satélite completo que está en estos momentos bajo desarrollo.

Tras la finalización de su desarrollo, la propiedad de EGNOS fue transferida a la Comisión Europea el 01/04/2009 y en estos momentos la operación de sistema es responsabilidad de la Comisión Europea a través de un contrato con el ESSP (European Satellite Service Provider).

EGNOS es un servicio abierto desde el 01/10/2009 y la información de posicionamiento está disponible libremente en Europa a través de señales satélite para cualquiera que posea un receptor GPS compatible con EGNOS.

Tras la certificación de EGNOS, la funcionalidad SoL (Safety of Live) fue declarada oficialmente disponible para aviación el 02/03/2011. Las señales de navegación EGNOS son utilizables para tareas críticas de seguridad en el guiado de aeronaves (vertical y horizontalmente) durante las maniobras de aproximación.

2.2 LAS ESTACIONES RIMS

Las estaciones del segmento de tierra o RIMS (Range and Integrity Monitoring System) desplegadas para supervisar los satélites de las constelaciones GNSS. Cada satélite tiene que ser supervisado por múltiples RIMS antes de que se generen las correcciones y los mensajes de integridad. El siguiente esquema del sistema EGNOS permite ver la situación de las RIMS dentro del sistema:

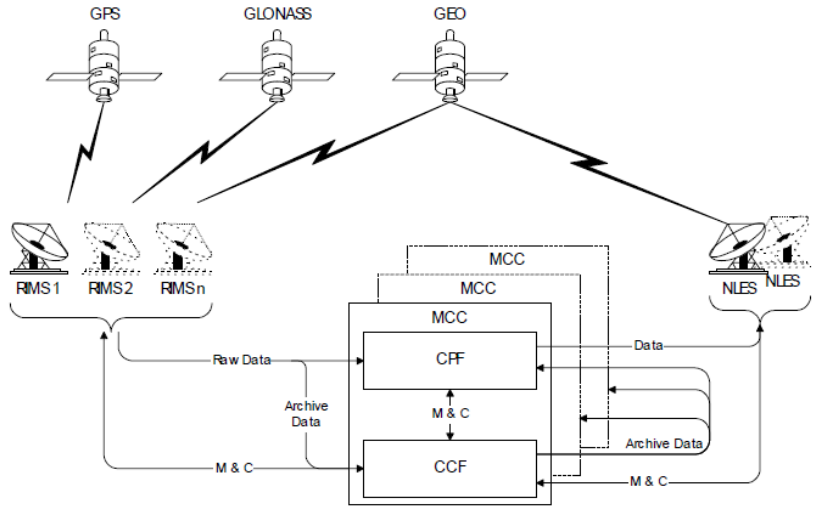


Figura 1 – Composición del sistema EGNOS

Las estaciones RIMS desplegadas se muestran en la siguiente figura representadas por un círculo:

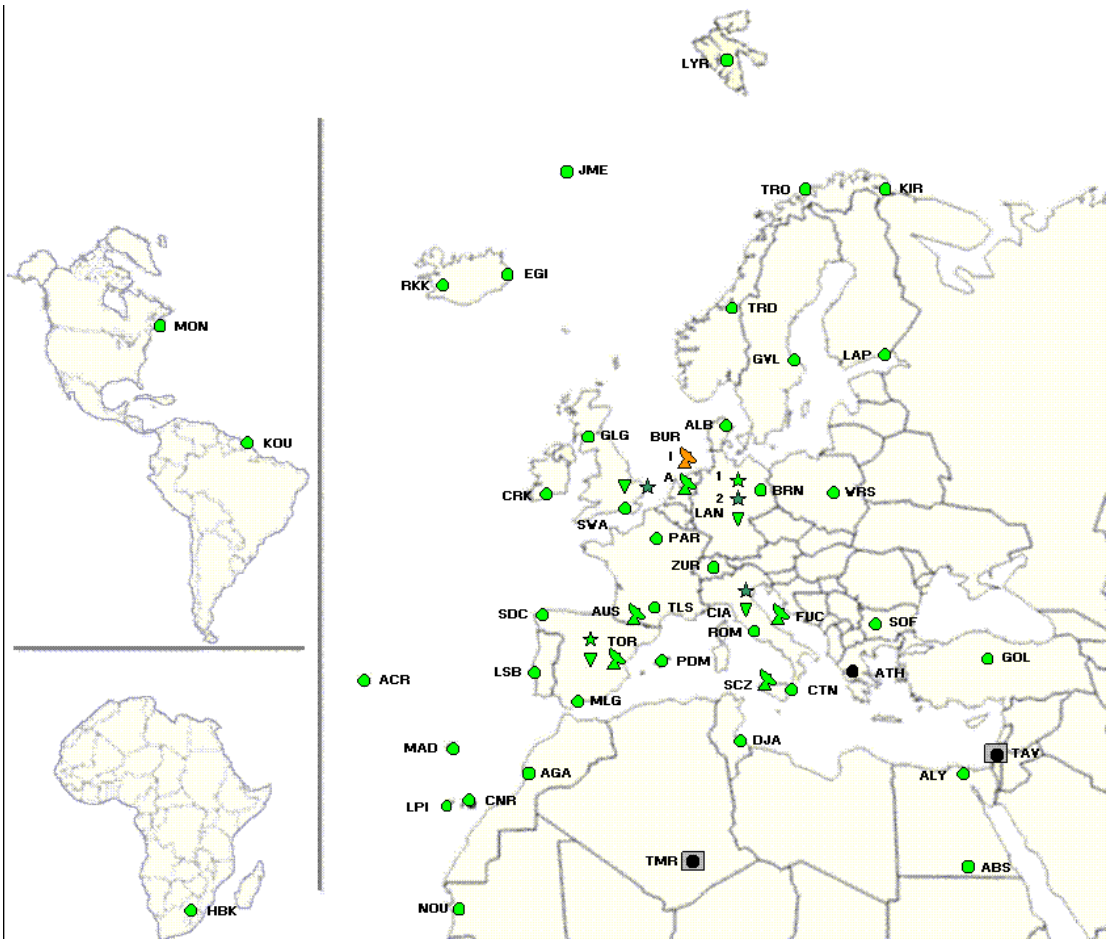


Figura 2 – Disposición de las estaciones del segmento de tierra de EGNOS

Cada estación RIMS tiene 3 partes principales:

- Receptor GPS de altas prestaciones: Receptor profesional capaz de recibir y procesar las señales GPS, GLONASS y de los satélites geoestacionarios que forman parte del sistema EGNOS.

- Referencia frecuencial: reloj atómico que genera una señal de 10 MHz que es usada por el receptor de altas prestaciones.

- Core Computer (CC): Computador industrial sobre el que corre la aplicación encargada de gestionar todos los equipos de la estación y de comunicarse con el centro de control. La aplicación envía 2 tipos de mensajes al centro de control: mensajes de monitorización y mensajes de RAW data que contienen la información transmitida por los satélites. Los mensajes de RAW_data tienen unas restricciones temporales muy grandes (por eso se usa un sistema operativo de tiempo real).

El diagrama de bloques con los equipos que forman una estación RIMS A puede verse en el diagrama siguiente:

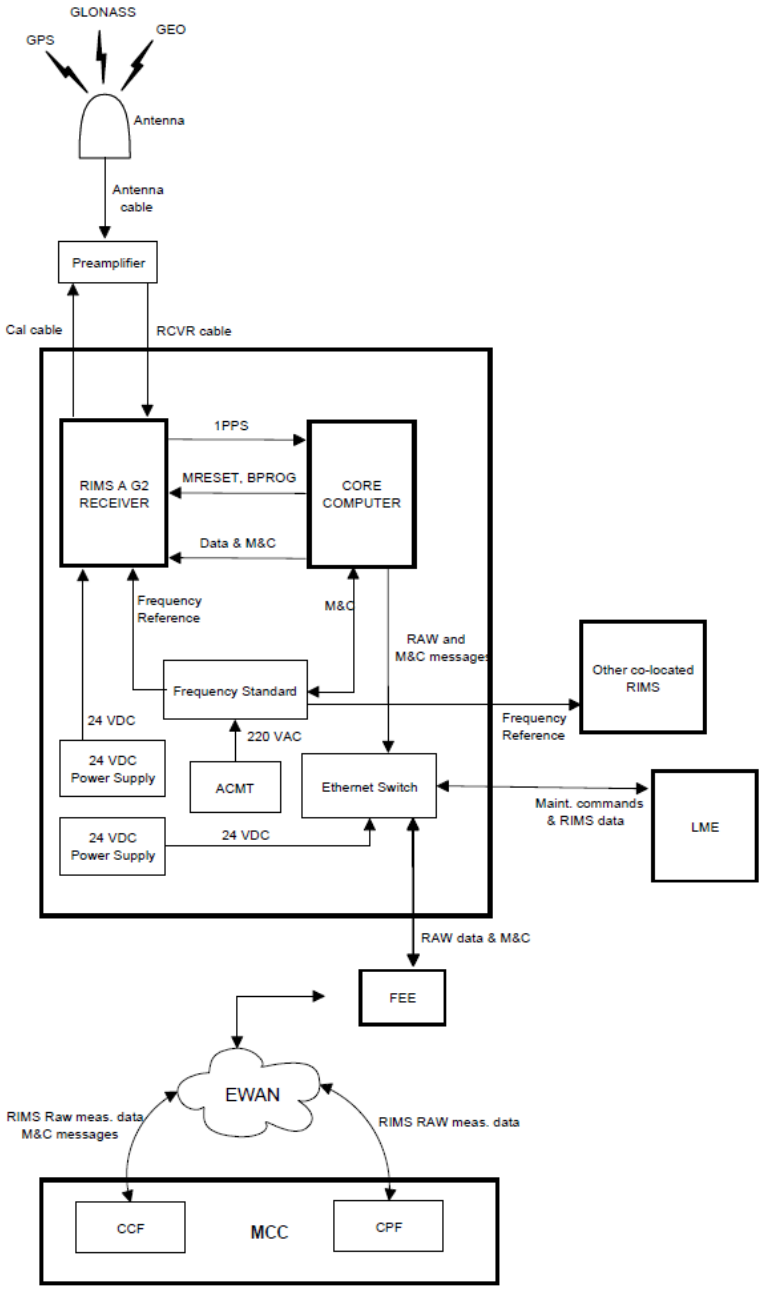


Figura 3 - Diagrama de bloques de la estación RIMS

2.3 EL CORE COMPUTER DE LA RIMS

En Core Computer (CC) de la RIMS es un computador industrial de altas prestaciones sobre el que corre la aplicación que controla la estación sobre un sistema operativo en tiempo real, en nuestro caso, LynxOS 3.1.0 de Lynuxworks.

El Core Computer tiene 2 objetivos fundamentales:

- Proporcionar al CCF (Central Control Facility) la funcionalidad de Monitoring&Control sobre la estación. Es decir, el CCF ejecuta los comandos que el CCF le ordene para operar la estación y proporciona una vez por segundo un mensaje de monitorización en el que se indica el estado de los diversos elementos HW y SW de los que consta la estación.

- Proporcionar al MCF (Master Control Facility) la información obtenida de los satélites GPS, GEO y GLONASS en tiempo real en cada época. En menos de 300 ms desde que comienza la época GPS (cada segundo), el CC debe de enviar un mensaje con toda la información obtenida de los satélites y proporcionada por el receptor GPS al MCF. Se requiere un sistema operativo en tiempo real y un gran control de las prioridades de los procesos para poder garantizar esta funcionalidad.

El diseño HW del CC es analizado en profundidad en capítulos posteriores, pero básicamente está formado por un chassis de altas prestaciones (puede embarcarse por ejemplo en helicópteros de combate), una placa base de alto rendimiento, una tarjeta PCI que nos proporciona 4 puestos serie RS-422 adicionales y un disco duro.

Debido a la obsolescencia del hardware del CC, y en particular de la placa base VMPC6a, es necesario migrar la aplicación del CC a una nueva plataforma HW. Con el fin de modificar lo menos posible la aplicación, se ha decidido modificar la BSP de la placa nueva (VM6250 de Kontron) para que el SO que usábamos en la plataforma anterior (LynxOS3.1) pudiera correr sobre la nueva placa.

2.4 OBJETIVOS DEL TFC

Como indica el título del TFC, este consiste es el análisis del proceso de migración de una aplicación embebida en tiempo real (la aplicación que corre en el CC), para una estación del segmento de tierra del sistema EGNOS (en nuestro caso, la estación RIMS A).

Más concretamente, el TFC propuesto se plantea los siguientes objetivos:

- Entender la problemática derivada del cambio de plataforma hardware para una aplicación embebida-

- Entender la funcionalidad y la importancia de la BSP

- Entender el proceso de validación software para un sistema embebido de tiempo real con nivel de criticidad DAL-C.

- Entender el proceso y la problemática derivada de la cualificación de sistemas DAL-C

- Describir los aspectos relacionados puramente con el hardware como las actividades de marcado CE.

Por evidentes motivos de confidencialidad, no se proporcionará código fuente y se omitirán algunos aspectos confidenciales del la migración, pero se describirá el proceso en profundidad.

2.5 LISTA DE TAREAS

A continuación, se describen brevemente las tareas en las que se divide el proyecto.

2.5.1 Diseño HW

- Adaptación del diseño del Core Computer: La plataforma está basado en un COTS (Customer On The Self) estándar que requiere algunas modificaciones (cableado interno, LEDs…).

- EL resto de la estación (excepto el Ethernet Switch que también está obsoleto) está basada en la misma plataforma que la estación actual. Esta actividad cubre:

    - Revisión del diseño para ver los componentes que deben comprarse.

    - Diseño un Retrofit Kit para el Ethernet Switch.

El output de esta actividad es la lista de componentes que deben comprarse o fabricarse.

2.5.2 Adquisición de componentes

Algunos de los components que deben comprarse tienen un plazo de entrega muy largo. Estos componentes son: el cabinet (se fabrica bajo pedido), el Core Computer (chasis y placa base) y tiene un plazo de entrega de unas 12 semanas.

2.5.3 DIseño detallado

Esta actividad incluye las siguientes tareas:

- Análisis de las diferencias entre la placa VM6250 (nueva) y la VMPC6a (obsoleta)

- Análisis del software actual, revisando los aspectos dependientes del hardware. Identificación de las limitaciones y posibles cambios a realizar en el código.

- Análisis del código fuente de la BSP de LynxOS 3.1 para VMPC5a y diseño del código de la BSP de LynxOS3.1 para VM6250.

- Actualización del diseño del Core Computer incluyendo: diseño, desarrollo, compra, product assurance, software, documentación de operación e instalación de acuerdo con la nueva configuración.

- Actividades de RAMS (Reliability, Availability, Maintainability and Safety)

2.5.4 Migración de la BSP de acuerdo a las necesidades del CC SW

Esta actividad será realizada con el apoyo de una empresa especializada en esta tarea. Las modificaciones serán testadas contra su diseño. En esta fase, se iniciará además la actividad de test de la API del SO (ver sección).

2.5.5 Adaptación del SW & AIV (Assembly, Integration & Validation)

Esta actividad incluye las siguientes tareas:

- Modificación del código fuente de la aplicación
- Ejecución de los tests de CC SW
- Ejecución de los tests de validación. Estos tests incluyen todos los tests a nivel de subsistema para garantizar el su buen funcionamiento
- Preparación de los informes de test asociados
- En esta fase, el SW está listo para ser integrado en la estación y las actividades de marcado CE y los test de non regression pueden ser ejecutados.

2.5.6 Marcado CE

Las actividades de marcado CE se realizarán para el core computer individualmente y para la estación completa.

Las actividades de marcado se realizarán en un laboratorio certificador.

2.5.7 Tests de Non-Regression

Esta sección incluye la ejecución de los tests de non-regression y la preparación de la documentación asociada.

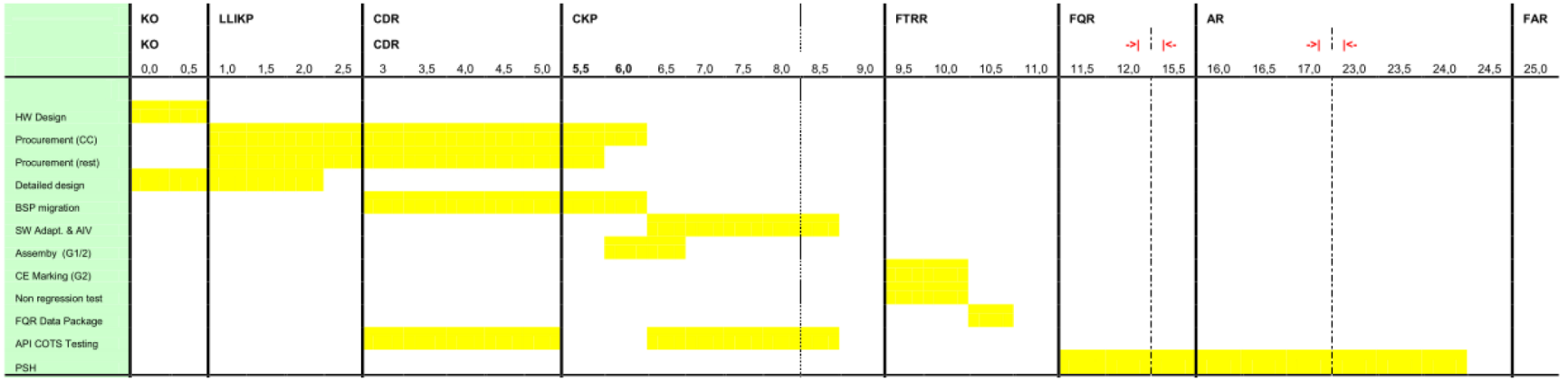
2.5.8 Actividades de cualificación de los COTS

Para cualificar los COTS de software, se realizarán las siguientes 2 actividades:

- Tests sobre la API del SO LynxOS 3.1 (esta actividad será realizada por una empresa especializada externa)
- Actividades de certificación por PSH (Product Service History) sobre la estación completa en un entorno representativo durante 1 año.

2.6 PLANIFICACIÓN

La planificación del proyecto, acordada con el cliente se realiza según el estándar ESA, en el que se definen una serie de hitos asociados a entregas de documentación y hardware. La siguiente figura muestra el diagrama de Gantt con la planificación y el listado de hitos del proyecto. En el momento de entregar esta memoria, estamos a punto de alcanzar el FTRR.



Acrónimo de los hitos	Nombre	Fecha
KO	Kick Off	T_0
LLIKP	Long Lead Item Key Point	T_0 + 1 mes
CDR	Critical Design Review	T_0 + 3 meses
CKP	COTS Key Point	T_0 + 5.5 meses
FTRR	Factory Test Readiness Review	T_0 + 9.5 meses
FQR	Factory Qualification Review	T_0 + 11.5 meses
AR	Acceptance Review	T_0 + 16 meses
FAR	Factory Acceptance Review	T_0 + 25 meses

Figura 4 – Planificación del proyecto

3. DISEÑO DETALLADO

3.1 ANÁLISIS HW/SW

Para poder asegurar la migración dentro de una planificación tan apretada, se tuvieron en cuenta los siguientes factores:

- Arquitectura seleccionada: La placa VM6250 ha sido diseñada por Kontron como la evolución natural de la placa VMPC6a (actualmente usada en el sistema). Por lo tanto, son altamente compatibles excepto para aquellos devices nuevos como por ejemplo el controlador Ethernet. El uso de esta arquitectura minimiza los cambios necesarios a nivel de BSP, CSP y toolchain.

- Uso del Advanced Porting Kit: LynxOS ofrece un diseño modular formado por los siguientes componentes: kernel, BSP, CSP, DRM y drivers. Los cambios derivados del hardware se limitan a la BSP, CSP y los drivers. LynuxWorks proporciona una herramienta llamada Advanced Porting Kit para migrar el SO hacia nuevas plataformas hardware.

- Prototyping: Antes de comenzar con el proceso oficial de migración, se preparó un prototipo en el que se consiguió que una primera versión de la BSP corriera sobre la placa VM6250 con las siguientes funcionalidades:

    - Driver serie (TTY)
    - Board and System reboot
    - Driver de red (TSEC)
    - GPIO
    - Watchdog

- Reusabilidad: Debido la compatibilidad a nivel HW y confirmado mediante el prototipo, existe un alto nivel de reusabilidad del SW.

3.1.1 Arquitectura hardware

La placa VM6250 fue seleccionada por tener los mayores niveles de compatibilidad con la placa VMPC6a actual. Las principales diferencias se resumen en la siguiente tabla:

Elemento	VMPC6a	VM6250	Comentarios
CPU	PowerPC 750L 450 MHz	MPC8640 1GHz	
RAM	512 MB EEC	1 GB DDR2	
HDD	73 GB SCSI	250 GB SATAII	

Ethernet (on-board)	DEC DC21143 PCI con 2x10/100 BASE T	BCM5466R PHY con 4x 10/100/1000 BASE TX	En la placa VM6250 es parte del procesador
Serial (on-board)	2x EIA-232	2x EIA-232	Integrado en el procesador
Serial (PMC)	TPMC866-11 (8x EIA-422)	TPMC866-11R (8x EIA-422)	Tarjetas completamente equivalentes
GPIO / Watchdog	6x GPIO pins + HW watchdog	3x GPIO pins + HW watchdog	

Tabla 1 - Diferencias entre las plataformas VMPC6a y VM6250

A continuación se resume el impacto de las diferencias mostradas en la tabla anterior:

- CPU: La nueva arquitectura es la evolución del PowerPC 750L usado en la placa VMPC6a. El repertorio de instrucciones y registros del nuevo procesador es un superset de los de la placa antigua, por lo tanto, no es necesario modificar la cadena de compilación ya que el código que será generado contiene instrucciones completamente soportadas por la placa VM6250. Las instrucciones avanzadas incluidas en el VM6250 no será utilizadas. Además, el CSP (Ship Support Package) de LynxOS, la parte del SO que trata los asuntos específicos del procesador, también se mantiene porque el nuevo es completamente compatible con el anterior.

- RAM: Sólo se requieren unos pequeños ajustes en la inicialización de memoria por parte de la nueva BSP debido a las diferencias en el mapa de memoria de la nueva placa.

- Disco duro: La placa VMPC6a usa un procesador SYM53C875A PCI-SCSI I/O para controlar el disco duro mientras que la placa VM6250 usa el controlador SATAII Sil3132 para controlar el disco duro. EL impacto de esta diferencia radica en que el driver SCSI será reemplazado por el driver SATA de LynxOS. Dicho driver deberá ser modificado desde nuevas versiones de LynxOS hasta el LynxOS 3.1.

- Ethernet: La nueva placa VM6250 usa un controlador Ethernet MPC8640 que forma parte del procesador junto con un BCM5466R PHY que está en la placa, por lo que es necesario creas nuevos driver para ambos dispositivos.

- Interfaz serie (on-board): Ambas placas usan una UART estándar , por lo que se puede usar el driver serie existente.

- Interfaz serie (placa PMC): La nueva placa TPMC866-11R es totalmente equivalente a la TPMC866-11, excepto por los materiales usados (la 11R cumple con RoHs), por lo que puede reutilizar el driver existente.

- GPIO + Watchdog: En ambos casos, las líneas GPIO y el watchdog son accesibles desde el procesador a través de una FPGA conectada el bus PCI, por lo tanto, sólo es necesario un pequeño driver para garantizar el acceso de la aplicación a los registros GPIO.

Las siguientes figuras muestran los esquemas de las arquitecturas de las placas VMPC6a y VM6250:

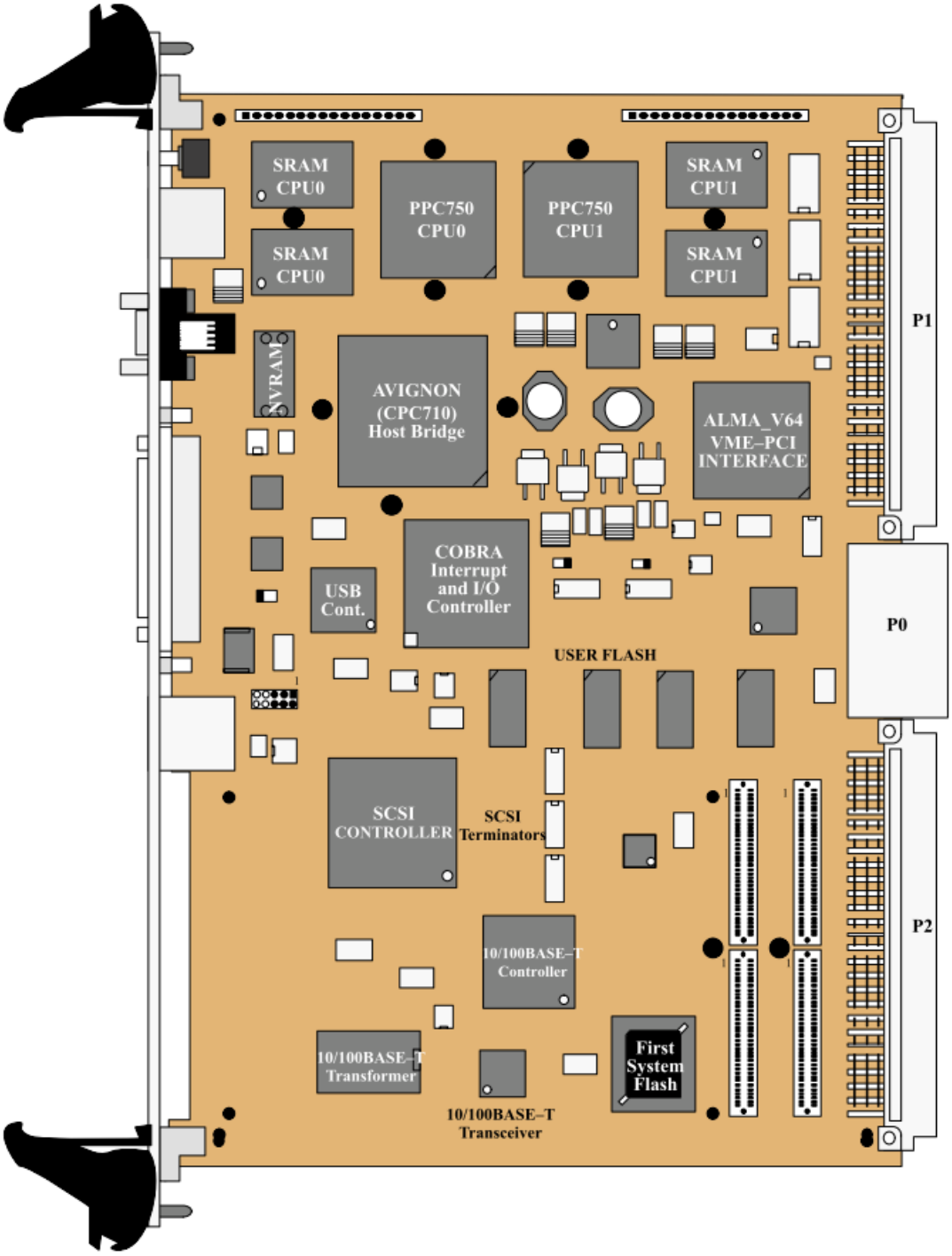


Figura 5- Arquitectura de la placa VMPC6a

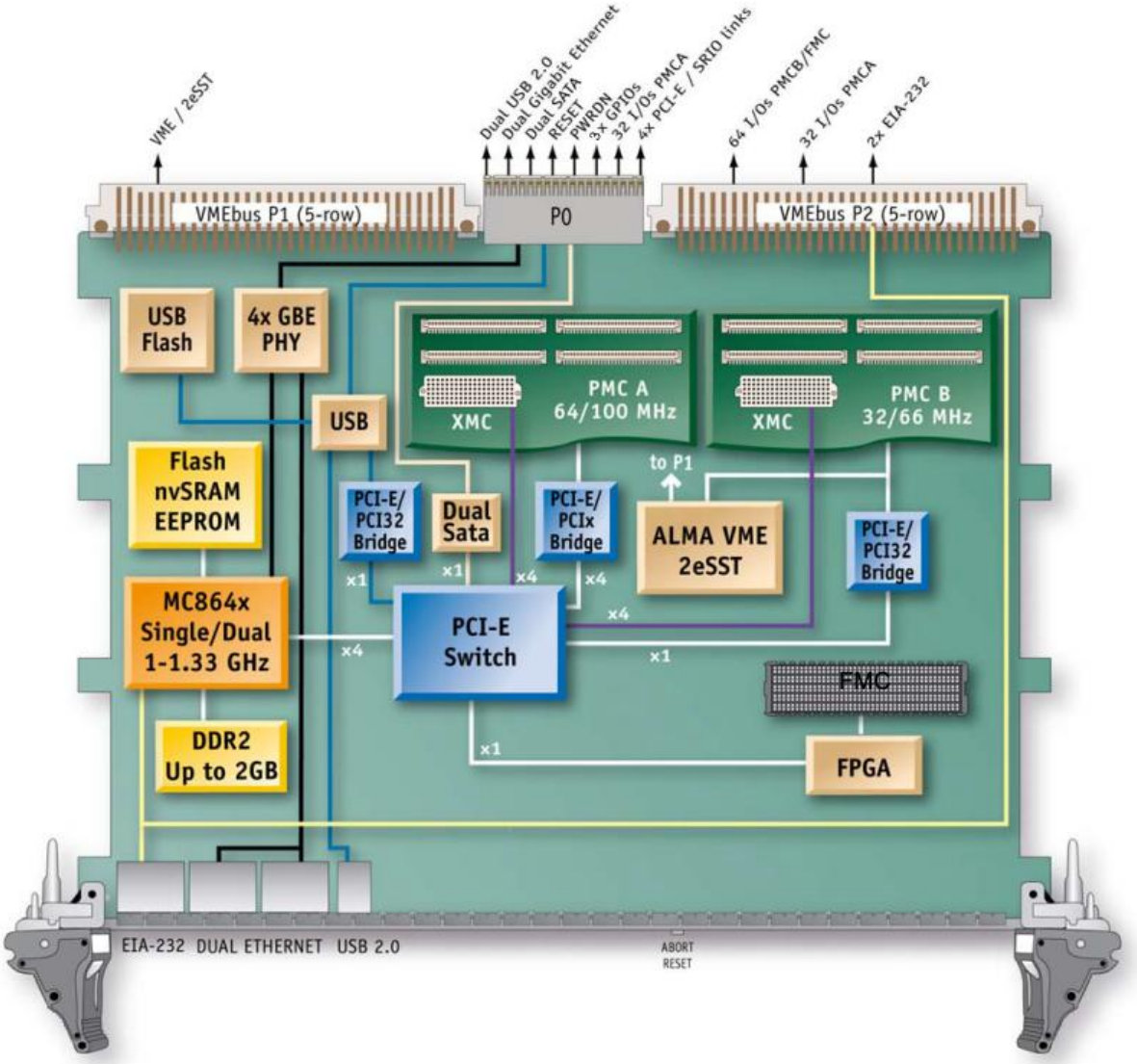


Figura 6 - Arquitectura de la placa VM6250

3.1.2 Análisis del sistema operativo (LynxOS 3.1.0a)

3.1.2.1 Funcionalidades del sistema operativo

LynxOS es un sistema operativo de tiempo real compatible con UNIX y que cumple con POSIX. Ha sido diseñado para ser usado en aplicaciones críticas donde es crucial tener una respuesta en un tiempo predecible. Sus principales funcionalidades son:

- Entorno multithread y multiproceso
- Manejo de la memoria a través de la MMU (Memory Management Unit) hardware

- Memoria virtual configurable
- Sistema de ficheros UNIX-like
- Arquitectura modular y escalable
- Kernel threads
- Network File System (NFS)
- TCP/IP stack estándar
- Remote procedure call (RCP)
- GNU tools y utilidades UNIX-like
- Kernel ROM-able

LynxOS tiene un diseño modular formado por los siguientes componentes:

- Kernel: Contiene aquellas partes del sistema operativo que son independientes del hardware. Básicamente son el scheduler (planificador) y los módulos encargados del manejo de la memoria. Esta parte es el nucleo del sistema y no cambia para adaptar el sistema a una nueva plataforma HW.

- CSP (Chip Support Package): Esta parte contiene el código del SO para una familia de CPUs en particular como por ejemplo la PowerPC. Esta parte cambia cuando hay un cambio significativo en el procesador de una plataforma a otra. 2 placas diferentes con el mismo procesador, o al menos con procesadores de la misma familia, puede usar el mismo CSP.

- BSP (Board Support Package): Todo el código necesario para manejar una placa específica se encuentra en este módulo. Esta parte es la responsable de iniciar el sistema al arrancar e interactúa, a través de interfaces simples con el kernel y el CSP. Esta parte contiene además los drivers de los dispositivos que son encargados de conectar la BSP con una parte o dispositivo específico de la placa (placa Ethernet, puerto serie, disco duro, bus VME…)

- DRM (Device Resource Manager): Este modulo es el responsable de la gestión PCI en caso de necesitar una manejo dinámico de dispositivos PCI. Si se usan drivers estáticos, este módulo no es necesario.

Todos estos componentes tienen interfaces con el resto y es posible combinar diferentes implementaciones de cada uno de ellos para formar una versión específica del sistema operativo.

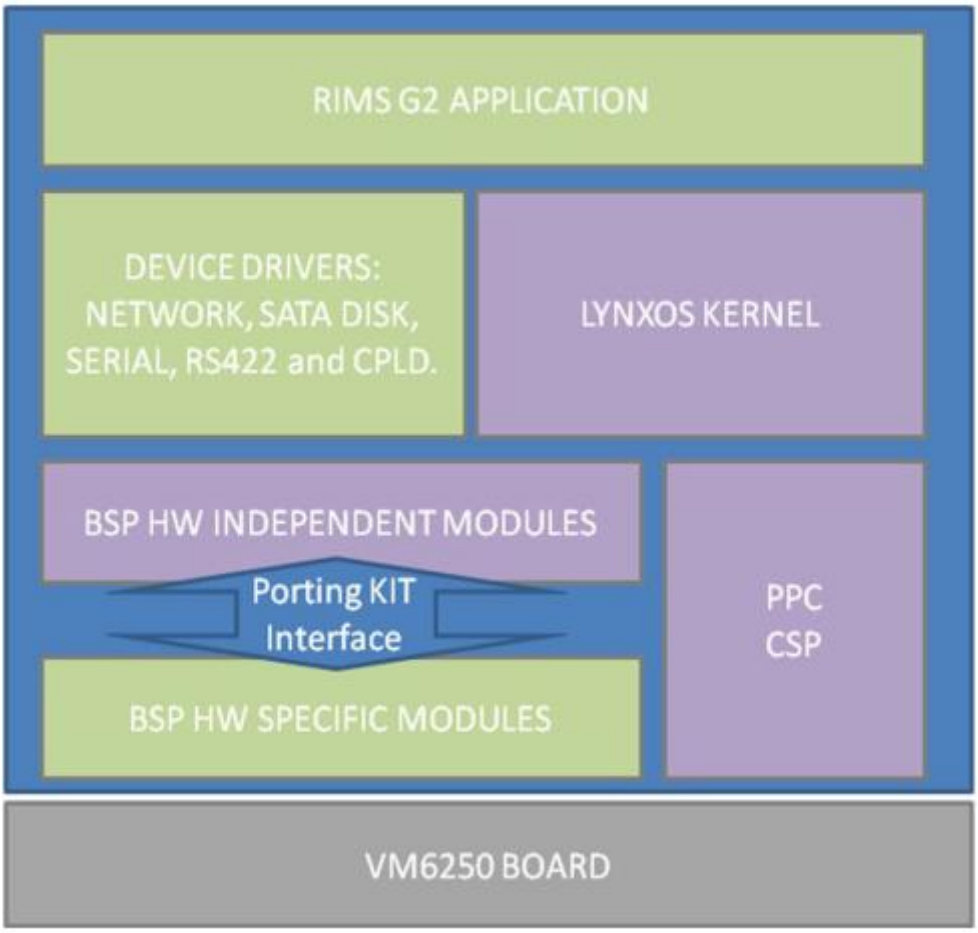


Figura 7 - Arquitectura software

Nuestro objetivo es describir el diseño de la nueva BSP que haga que el sistema operativo completo sea ejecutable en una placa VM6250. Esta arquitectura permite que migrando una pequeña parte del sistema operativo, el resto de los componentes del sistema operativo funcionen en la nueva plataforma. Además, varios drives son además necesarios para completar el acceso a los dispositivos de la placa (on-board).

3.1.2.2 Advanced Porting Kit

El Advanced Porting Kit es una herramienta que LinuxWorks proporciona a sus clientes para que permitir la migración del sistema operativo a placas que no están soportadas. El porting kit proporciona el kernel de la BSP y el código fuente de drivers correspondientes a plataformas soportadas que pueden ser usados como puntos de partida para la creación de nuevas BSPs.

El proceso de desarrollo de una nueva BSP para una nueva plataforma HW consiste en varios pasos caramente identificados en la documentación que se proporciona junto con el porting kit. Además, el porting kit proporciona una serie de funciones vacías que el desarrollador debe implementar con una funcionalidad perfectamente definida en la

documentación (cada parámetro, cada acción sobre el HW y el posible valor del retorno están definidos).

Junto con el porting kit, LynuxWorks proporciona el ATS (Automatic Test Suite) que puede ser usado por el desarrollador para escribir y ejecutar tests para validad la BSP creada.

3.1.2.3 Estructura de la BSP

Las BSPs de LynxOS se pueden dividir en 2 partes bien diferenciadas, una primera parte que trata directamente con el hardware y otra parte que se comunica con el kernel. La primera parte consiste en un conjunto de funciones que el desarrollador tiene que rellenar con código específico para el hardware mientras que la segunda parte permanece inalterada en la mayoría de los casos. Además, encontramos los drivers de dispositivos que tratan con drivers de periféricos conectados dentro de la placa. Estos driver contienen código dependiente del HW junto con un interfaz común a través del cual son utilizados por el kernel. A continuación se muestra la interacción de dichos módulos:

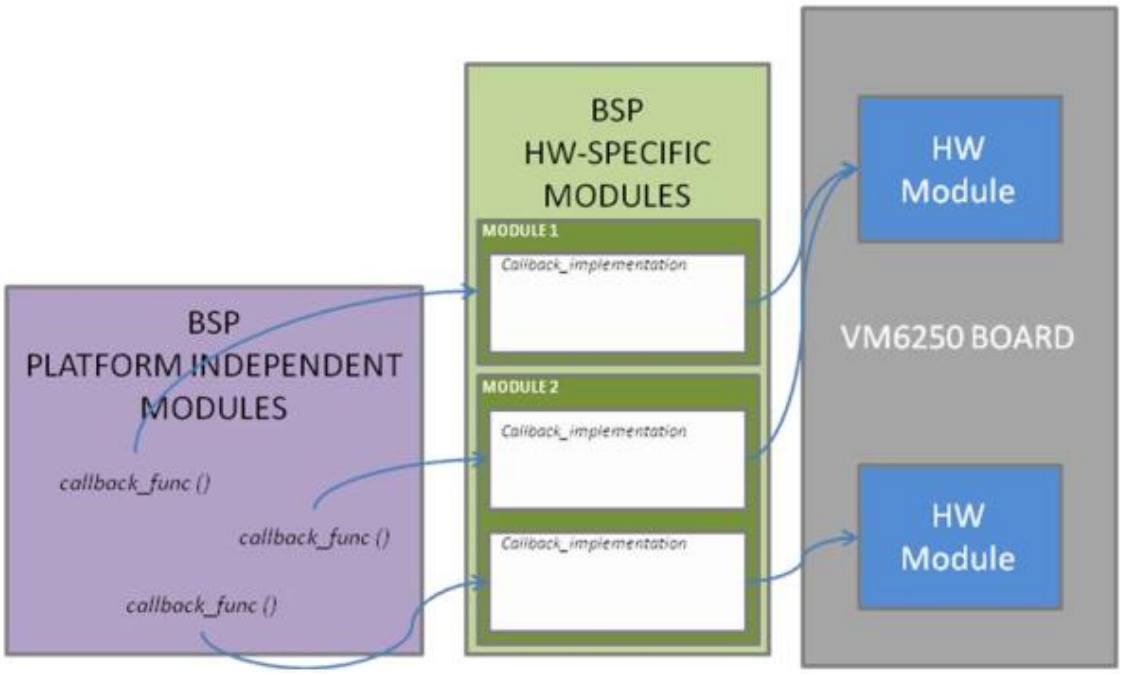


Figura 8 - Estructura de la BSP

3.1.2.4 Drivers de dispositivos en LynxOS

3.1.2.4.1 Estructura de los drivers de LynxOS

Los drivers de dispositivos de LynxOS proporcionan una interfaz consistente para el que el kernel solicite transferencias de datos hacia/desde el dispositivo y lo controle de la manera adecuada. La siguiente figura describe el modelo de driver de LynxOS:

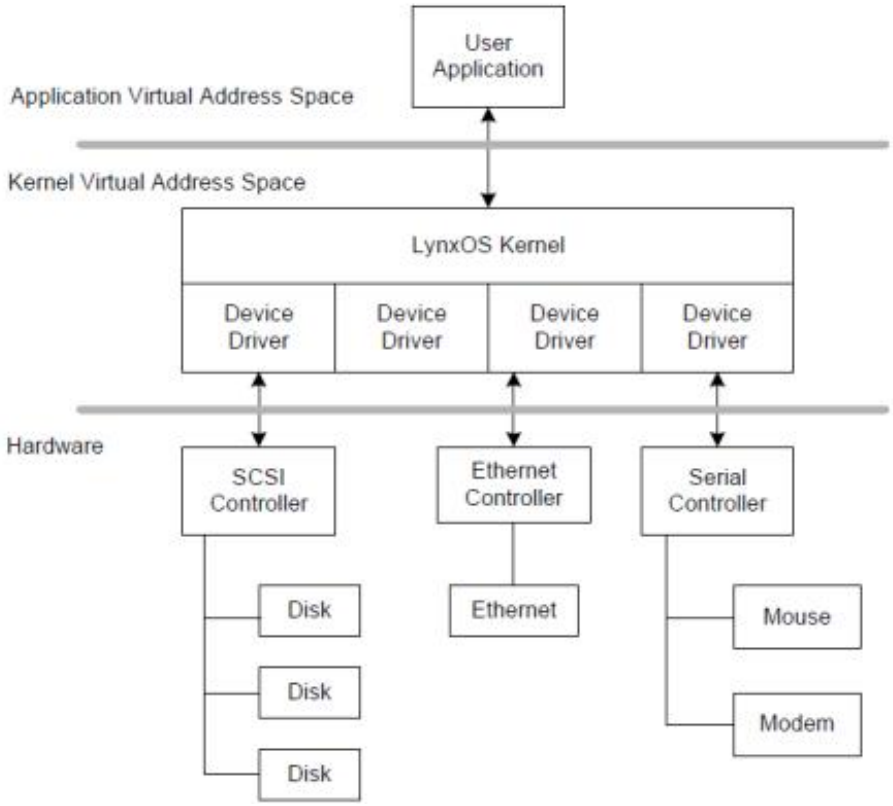


Figura 9 - Estructura de los drivers de LynxOS

LynxOS soporta 2 tipos de drivers en función de los requerimientos del dispositivo HW:

- Drivers de bloques: Las transferencias de datos se realizan utilizando un tamaño de datos fijo
- Drivers de caracteres: Las transferencias de datos tienen un tamaño variable

Los drivers consisten en funciones (funciones de entrada o entry functions, kernel threads y funciones auxiliares) y estructuras de datos.

3.1.2.4.2 Funciones de entrada (entry point functions)

Estas rutinas proporcionan un interfaz definido con el resto del kernel. El kernel llama a la entry point apropiada cuando la aplicación del usuario hace una llamada al sistema. Estas funciones quedan resumidas en la siguiente tabla:

Entry point	Descripción

Entry point	Descripción
install()	Inicializa el HW y reserva los recursos necesarios por el driver del dispositivo
uninstall()	Libera los recursos asignados durante install()
open()	Inicializa dispositivos menores
close()	Se llama sólo cuando el último descriptor de fichero abierto apuntando a un dispositivo menor es cerrado
read()	Lee data del dispositivo
write()	Envía data al dispositivo
ioctl()	Ejecuta un comando específico del dispositivo
select()	Soporta I/O polling o multiplexado
strategy()	Planifica una operación de lectura o escritura en un dispositivo de bloques
mmap()	Mapea datos a memoria

Tabla 2 - Entry point functions

3.1.2.4.3 Estructura de datos de dispositivo

La estructura de datos del dispositivo se usa para pasar parámetros específicos del HW al driver del dispositivo. Estos parámetros, típicamente de configuración, son esenciales para el adecuado funcionamiento del driver pero limitan el grado de aplicabilidad del driver si están hardcoded.

La estructura de datos del dispositivo existe dentro del espacio de direcciones del kernel. EL kernel pasa la dirección de la estructura de datos al entry point install().

3.1.2.4.4 Estructura de datos estática

Las rutinas dentro del driver deben operar sobre el mismo set de información para cada instancia del driver. En LynxOS, esta información se conoce como la estructura estática del driver.

El entry point install() devuelve el valor de la estructura estática y el resto de entry points reciben un puntero a la estructura estática por parte del kernel.

3.1.2.4.5 Gestión de memoria

LynxOS usa una arquitectura de memoria virtual. Todas las direcciones de memoria generadas por la CPU son traducidas por el hardware MMU (Memory Management Unit). Cada tarea de usuario tiene su propio espacio de memoria virtual protegida que evita que

las tareas interfieran entra si (consciente o inconscientemente). El kernel, que incluye los driver de dispositivos, y la tarea de usuario que se está ejecutado en ese momento comparten el mismo espacio de memoria virtual. La tarea de usuario se mapea en parte baja y el kernel en la parte alta. Sólo la parte ocupada por la tarea de usuario es remapeada durante el cambio de contexto.

Las aplicaciones no pueden acceder a la parte de la memoria ocupada por el kernel y sus estructuras de datos. La constante OSBASE define el límite superior de la memoria accesible. Las direcciones sobre este límite sólo pueden ser accedidas por el kernel.

3.1.2.4.6 Sincronización

Existen varios mecanismos de sincronización que pueden ser usados por un driver de dispositivo en LynxOS:

- Semáforos de kernel: un semáforo de kernel es simplemente una variable entera declarada por el driver. Todos los semáforos deben ser visibles en todos los contextos. Esto significa la memoria para un semáforo no debe ser asignada al stack. Un semáforo se adquiere mediante la función swait(). Cuando un semáforo se libera, si hay varias tareas esperando en la cola del semáforo, aquella con la prioridad más alta se hace con el semáforo.

- Desabilitación de interrupciones: Lynuxworks proporciona rutinas para habilitar/deshabilitar interrupciones. Las rutinas para controlar las interrupciones son disable() y restore().

- Disabling preemption

3.1.2.5 Manejo de interrupción y del timeout

Los manejadores de interrupciones en LynxOS son especificados en los entry points install() y open() y son desasignados por uninstall() y close(). Estos manejadores de interrupciones se inician antes que cualquier otro proceso del kernel o de la aplicación.

Dado que los manejadores de interrupciones tienen la prioridad más alta, minimizar la longitud de cada rutina de servicio de la interrupción es de gran importancia.

Hay diferencias significativas en el manejo de interrupciones entre el procesador de la placa VMPC6a y el de la placa VM6250. Con la BSP para x86, LynxOS utiliza una tabla proporcionada por la BIOS para conectar con las ISR (Interrupt Service Routine) de los drivers. La BIOS incluye código para manejar las líneas de interrupción de la placa y decidir qué ISR debe ser ejecutada.

Bajo una arquitectura canónica PowerPC, todo es manejado por las rutinas de inicialización del bootloader. Las arquitecturas PowerPC no tienen una BIOS similar a la arquitectura x86, el manejo de interrupciones se hace de una forma diferente y las interrupciones PCI se manejan a nivel de sistema operativo, por lo tanto, cualquier problema de inicialización encontrada en las BSPs x86 no existen por definición por definición en las BSPs PowerPC.

3.1.3 FUNDAMENTOS DE DISEÑO DE LA BSP DE LA PLACA VMPC6a

3.1.3.1 Visión general de la BSP de VMPC6a

La BSP de VMPC6a se proporciona en el Advanced Porting Kit junto con muchas otras BSPs para otras arquitecturas (Motorola Power Platform, Motorola Power Plus, Motorola CPCI, Cetia CVME, Force Powercore, PMC860, MBX860 y MPC860 FADS).

La placa VM6250 ha sido diseñada por Kontron como el sustituto natural de la obsoleta VMPC6a, por lo tanto, su compatibilidad es alta excepto para aquellos dispositivos que han sido renovados, como por ejemplo el controlador Ethernet.

Como el diseño de la BSP para VMPC6a no es proporcionado por el fabricante, se ha hecho un estudio detallado del código fuente para tener una idea clara de los fundamentos a la hora de construir la nueva BSP. Las indicaciones proporcionadas en la documentación del porting kit han sido muy útiles para navegar por el código fuente.

Las siguientes secciones analizan más detalladamente la BSP.

3.1.3.2 Estructura de la BSP para VMPC6a

La BSP para VMPC6a sigue la misma estructura de directorios que el resto de la BSPs tal y como se describe en el porting kit:

Elemento	Descripción
bsp.vmpc	Dierctorio que contiene el código principal de la BSP que es responsable de inicializar la placa y arrancar el kernel de LynxOS
drivers.vmpc	Directorio que contiene los drivers de los dispositivos de la placa. Contiene subcarpetas para cada uno de los drivers: - almavme: controlador del bus VME - if_dec21040: controlador Ethernet - ll-scsi: controlador del disco duro SCSI generico - simncr: modulo del controlador SCSI - pci: bus PCI - pwbflash: driver de la memoria flash - tty: controlador de los puertos serie de la placa
devices.vmpc	Esta carpeta contiene las estructuras de información para cada uno de los dispositivos

Tabla 3 - Componentes de la BSP para VMPC6a

3.1.3.3 Controladores de dispositivos de la BSP para VMPC6a

Los drivers implementados en la BSP no podrán ser reusados con excepción del los drivers tty (puertos serie) y el driver para la placa TPMC866-11R.

Estos drivers están programados para controlar dispositivos hardware específicos (VME, SCSI, tarjeta de red…) que no existen en la nueva placa VM6250. Además, los dispositivos equivalentes de la placa nueva son significativamente diferentes a los actuales (en interfaz de red es parte del procesador, se usa un interfaz SATA en vez de SCSi para el disco duro...).

3.1.4 NECESIDADES DE SOPORTE HARDWARE DE LA BSP PARA VM6250

Las necesidades de la nueva placa son básicamente derivadas de los dispositivos hardware que la nueva BSP debe reconocer y usar. Esto incluye el ser capar de arrancar la CPU, inicializar la memoria y todos los dispositivos de comunicaciones de la placa además de proporcionar los controladores para aquellos periféricos que son directa o indirectamente usados por la aplicación como el interfaz de red y el disco duro. La siguiente tabla resume el hardware que deberá ser soportado:

Elemento	VM6250	Implementación	Comentarios
CPU	MPC8640 1GHz/533MHz	CSP	
RAM	1GB DDR2	BSP	
HD	250 GB SATAII	Driver	
Ethrenet (en la placa)	Controlado por el procesador + BCM5466R PHY proporcionando 4x 10/100/1000 BASE TX	Driver	Parte del procesador
Serie (en la placa)	2 x EIA-232	Driver	Parte del procesador
Seria (placa PMC)	TPMC866-11R(8 x EIA-422)	Driver	
GPIO / Watchdog	3 x GPIO pins + watchdog HW	Driver	

Tabla 4 - Necesidades de soporte HW para VM6250

Por lo tanto, desde el punto de vista de las interfaces que deben ser implementadas o usadas, y la forma en que son implementadas o usadas por la BSP y los controladores de dispositivos, se pueden resumir en la siguiente tabla:

Desde	Hacia	Detalles de la interfaz
Aplicación	Driver del Watchdog	Acceso directo a través de ioctl
Aplicación	Driver GPIO	Acceso directo a través de ioctl
Aplicación	Driver placa TPMC866-11R	A través del kernel (POSIX)
Aplicación	Driver TTY	A través del kernel (POSIX)
Aplicación	Driver SATA	A través del kernel (POSIX)
Aplicación	Driver de red	A través del kernel (POSIX)
CSP	Kernel	Interfaz interno
BSP	Kernel	Interfaz interno
BSP	Hardware	Funciones callback
Driver del Watchdog	BSP	Mapa de memoria
Driver GPIO	BSP	Mapa de memoria
Driver placa TPMC866-11R	BSP	Mapa de memoria / PCI bus
Driver TTY	BSP	Mapa de memoria
Driver SATA	BSP	Mapa de memoria
Driver de red	BSP	Mapa de memoria

Tabla 5 - Interfaces a implementar

3.1.4.1 REUSABILIDAD

La mayoría del sistema operativo de la plataforma obsoleta es reusado. Todo el kernel (planificador, threads, colas…), herramientas, librerías, CSO, DRM y la cadena de compilación se reusan completamente (confirmado por el portotipo).

Tal y como se ha descrito en secciones anteriores, únicamente una serie de archivos estrechamente ligados al harware requieren ser modificados o creados. Algunas consideraciones importantes con las siguientes:

- El driver SATA no se escribe desde cero, sino que se modificará el driver SATA existente para otras versiones de LynxOS como la 178.

- El driver de red no se escribe desde cero, ya que al estar integrado en el procesador, existe un driver hecho por el fabricante que puede ser modificado.

La tabla siguiente resume las partes del SO que deberán ser modificadas:

Elemento	Archivo	SLoC	Reuso	Comentario
BSP	version.c	18	50%	Implementado en el prototipo
BSP	hw_conf.c	34	75%	Específico del HW
BSP	hw_init.c	1410	75%	Específico del HW
BSP	hw_pci.c	321	75%	Específico del HW
BSP	hw_pcibus.c	509	75%	Específico del HW
BSP	hw_time.c	275	75%	Específico del HW
BSP	kput_ppc.c	¿?	0%	Implementado en el prototipo
BSP	hw_absolute.h	131	75%	Específico del HW
BSP	hw_conf.h	34	75%	Específico del HW
BSP	hw_pcibus.h	32	75%	Específico del HW
BSP	hw_proto.h	68	75%	Específico del HW
DRIVERS	GPIO/WD	¿?	0%	Implementado en el prototipo
DRIVERS	ETHERNET	¿?	0%	Basado en un driver existente
DRIVERS	TTY	650	90%	Implementado en el

				prototipo
DRIVERS	SATA	¿?	0%	Basado en un driver existente

Tabla 6 - Partes del SO a modificar

3.1.4.2 PROCESO DE ARRANQUE DE LA PLACA VM6250

El código de arranque (boot code) de la BSP para VM6250 es la parte más importante de la BSP (junto con los drivers).

El proceso de arranque consiste básicamente en los siguientes pasos:

- Arranque de la imagen del kernel: tras la creación del KDI, su imagen binaria es cargada en memoria por el u-boot (bootup monitor)
- Inicialización del mapa de memoria:
    - rkboot.c receta la CPU a un estado conocido apagando la MMU, deshabilitando la caché e inicializando el stack pointer. Las variables globales no son accesibles porque la MMU está apagada.
    - mmunit() inicializa las tablas de páginas de la MMU
- Inicialización de las cachés L1 y L2.
- Inicialización del controlador de interrupciones
- Otras tareas: inicializar el bus de errores, fecha y hora…
- Inicializar el kernel de LynxOS
- Inicializar la aplicación de usuario

3.1.4.3 DRIVERS DE DISPOSITIVOS PARA VM6250

La siguiente tabla resume los driver que deben ser implementados dentro de la BSP para VM6250.

Componente	Tipo	Kernel threads
Líneas serie	Driver	Máximo 4 threads (2 para cada canal RX/TX) Prioridad inicial: 17
Ethernet	Driver	Máximo 4 threads (uno para cada dispositivo) Prioridad inicial: 18

Disco SATA	Driver	1 kernel thread
GPIO	Driver	No
Watchdog	Driver	No
Placa PMC866	Driver	Similar al TTY: 2 kernel threads por cada canal RX/TX Prioridad inicial: 17

Tabla 7 - Drivers a implementar para VM6250

3.1.4.3.1 Puertos serie (TTY)

La placa VM6250 tiene un dispositivo serie compatible con el de la placa VMPC6a (16450 UART y PC16550D), por lo tanto, el driver se reutilizará de la BSP de la placa antigua. Sólo habrá que actualizar las direcciones de memoria de al UART de la placa VM6250, por lo que los cambios son mínimos.

3.1.4.3.2 CPLD (Watchdog y GPIO)

La arquitectura de este driver se reusará pero deberá ser fuertemente modificada, ya que tanto las funcionalidades de watchdog como la de GPIO son proporcionadas por distintos dispositivos en ambas placas. Los dispositivos se mapean en el bus PCI en ambos casos, por lo que una arquitectura simple basada en acceso directo a los registros del watchdog/GPIO utilizando ioctl que se utilizaba en la placa VMPC6a, es válida para la nueva placa.

3.1.4.3.3 Interfaz de red

La arquitectura del driver de red es completamente diferente en ambas placas. Por lo tanto, un nuevo driver será desarrollado específicamente pata la placa VM6250 soportando todos los interfaces de red disponibles. El driver Ethernet no proporciona una API accesible directamente por la aplicación, sino que es accedido a través de los servicios de red de LynxOS.

3.1.4.3.4 Driver SATA

Se debe implementar un nuevo driver para el controlador SATA Sil3132. El driver deberá soportar tanto las transferencias en modo bloque como el modo carácter. Los device nodes se crearán bajo la carpeta /dev.

3.1.4.3.5 Driver de la placa TEWS TPMC866-11R

La placa TEWS es una placa PMC que se conecta a la placa base proporcionando los puestos RS-422 que la aplicación requiere para la interconexión del Core Computer con el receptor GPS. La misma placa se utiliza tanto para la placa VMPC8a como la VM6250. Como utilizaremos el mismo hardware, no se espera que sea necesario hacer ningún cambio en este driver.

3.2 INTEGRACIÓN HARDWARE DEL CORE COMPUTER

En paralelo a la modificación de la BSP, se procedió a realizar el ensamblado de los distintos componentes del Core Computer y a analizar las modificaciones necesarias con respecto al Core Computer anterior. Como veremos más adelante, algunas de estas modificaciones tendrían un impacto sobre la migración de la aplicación siendo necesario realizar algunos cambios en el código fuentes para adaptarnos a dichos cambios.

El Core Computer definitivo está compuesto por los distintos componentes hardware:

Elemento	Fabricante	Modelo	Descripción
Chassis	Kontron	R2U4SV	Chassis en el que van instalados todos los componentes
Placa base	Kontron	VM6250	Componentes principal de Core Computer
Rear Tarnsition module (RTM)	Kontron	PBV36	Placa de expansión de puertos de la placa base
Placa serie	TEWS	TPMC866-11R	Proporciona 8 puertos RS-422 a través de un conector de 50 pines
Disco duro (SATA)	Western Digital	WD2503ABYX	Disco duro de altas prestaciones usado normalmente en servidores

Tabla 8 – Principales componentes hardware del Core Computer

3.2.1 Integración hardware del Core Computer

La tabla siguiente resume todos los interfaces que necesita el Core Computer y compara entre la forma en que estos interfaces se implementaban en el Core Computer antiguo y en el nuevo.

	VMPC6A	VM6250+R2U4S chassis

Interface	SW driver	Device	Connector Type	Connector location	Signal type	SW driver	Device	Connector Type	Connector location	Signal type	Notes
Atomic Clock M&C	serial	com 2	internally cabled to chassis DB9-Male	RTM	serial 232	serial	com 2	internally cabled to chassis DB9-Male	RTM-H8	serial 232	Cabled to RTM com2 connector
1 PPSCore RS422 from RX	tpmc866	TPMC866 1-8	LRU-19 homemade cable	TEWS board	TTL Input to Core	tpmc866	TPMC866 1-8	homemade cable	TEWS board		
1 PPSCore from RX	ctrl drvr	ctrl dev	DB9-Male (PIN TBD)	Core chassis.	TTL Input to Core	not used	not used				
MRESET to RX	ctrl drvr	ctrl dev	DB9-Male (PIN TBD)	Core chassis.	TTL output from Core	GPIO	GPIO1				
BPROG to RX	ctrl drvr	ctrl dev	DB9-Male (PIN TBD)	Core chassis.	TTL output from Core	GPIO	GPIO2				
422a to the RX-GPS	tpmc866	TPMC866 1-8	LRU-19 homemade cable	TEWS board	serial 422	tpmc866	TPMC866 1-8	homemade cable	TEWS board	serial 422	
422b to the RX-GGC	tpmc866	TPMC866 1-8	LRU-19 homemade cable	TEWS board	serial 422	tpmc866	TPMC866 1-8	homemade cable	TEWS board	serial 422	
RIMS data flow	if_dec21040.obj	dec	RJ45-Fem	OnBoard	Ethernet 10Mb	ehternet	eth0	RJ45-Fem	OnBoard	Ethernet	
Power											

	VMPC6A					VM6250+R2U4S chassis					
Interface	**SW driver**	**Device**	**Connector Type**	**Connector location**	**Signal type**	**SW driver**	**Device**	**Connector Type**	**Connector location**	**Signal type**	**Notes**
LED1	ctrl drvr	ctrl dev	Internally cabled to LED board	RTM	TTL output from Core	tpmc866	TPMC866 1-8				
LED2	ctrl drvr	ctrl dev	Internally cabled to LED board	RTM	TTL output from Core	tpmc866	TPMC866 1-8				
LED3	ctrl drvr	ctrl dev	Internally cabled to LED board	RTM	TTL output from Core	tpmc866	TPMC866 18-				
LED4	ctrl drvr	ctrl dev	Internally cabled to LED board	RTM	TTL output from Core	Not available					
Console	serial	com 1	RJ12-Fem	OnBoard	RS232	serial	com 1	RJ12-Fem	OnBoard	RS232	

Tabla 9 – Comparativa de interfaces

A partir de la tabla anterior podemos identificar los siguientes cambios:

- El CC antiguo tenía numerosas señales GPIO en la placa VMPC6a con la que podíamos controlar los LEDs y los interfaces TTL con el receptor, mientras que la nueva placa sólo dispone de 3 señales GPIO, inicialmente pensadas para controlar los 3 LEDs de status del chassis.

- El CC antiguo tenía 4 LEDs de estado en el chassis mientras que el nuevo tiene sólo 3.

3.2.2 Modificaciones hardware

Como explicamos en el punto anterior, las principales modificaciones necesarias están relacionadas con la nueva forma de gestionar los LEDs. Además de esta modificación, es necesario que los puertos serie sean accesibles desde el exterior de chassis, para lo que tenemos que ubicar conectores adicionales en el chassis así como diseñar un nuevo cableado interno.

3.2.2.1 Gestión de LEDs

Al estudiar la forma de gestionar los LEDs de estado nos encontramos con el problema de que no teníamos suficientes señales GPIO disponibles, ya que 2 de las 3 se destinaron a señales de control del receptor.

La siguiente imagen muestra la situación de los LEDs en el frontal de chassis del Core:

Figura 10 - LEDs de estado del Nuevo chassis (derecha)

La solución alcanzada consistió en controlar los LEDs utilizando 3 de los 5 puertos RS422 libres de la plata TEWS. Los 8 puertos de la placa se gestionan mediante 2 UARTS de 4 puertos cada una. Los puertos 0, 1 y 2 se utilizan para la comunicación con el receptor y todos ellos pertenecen a la misma UART. Para aislar lo máximo posible la comunicación con el Rx de la gestión de los LEDS, utilizaremos la UART libre para gestionarlos:

- Puerto 4: LED naranja
- Puerto 5: LED rojo
- Puerto 6: LED verde

El esquema eléctrico resultante se muestra en la imagen inferior:

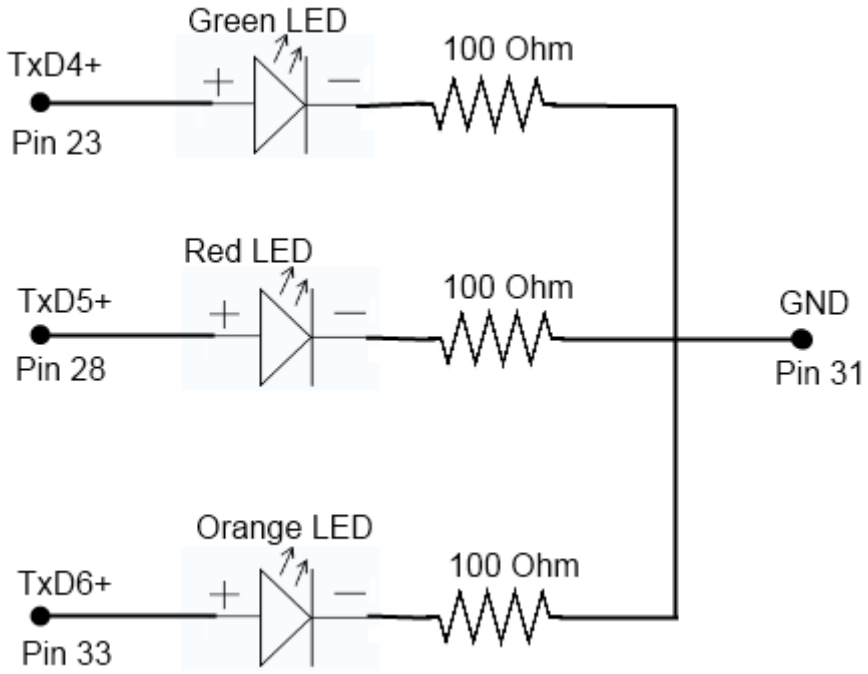


Figura 11 - Esquema eléctrico de los LEDs de estado

La placa TMPC866-11R proporciona 8 puertos RS-422. RS-422 utiliza señales en modo diferencial, esto es, el voltaje de una señal para saber si está a nivel alto o bajo no se mide entre dicha señal y tierra sino entre los 2 pines asociados a la señal (por ejemplo Tx+ y Tx-) que en esta placa en concreto corresponde al mismo voltaje que entre Tx+ y tierra.

Cuando un puerto está inactivo (sin transmisión), Tx+ está puesta a nivel alto. El nivel de Tx+ cuando el puerto no está transmitiendo se controla a través de la señal BREAK. SI BREAK está activada, Tx+ se pone a nivel bajo, y cuando se desactiva, pone Tx+ a nivel alto. Esto se realiza a través de la llamada al sistema `ioctl`.

- `ioctl(fd, TIOCSBRK, (char*)0)` Activa la señal BREAK
- `ioctl(fd, TIOCCBRK, (char*)0)` Desactiva la señal BREAK

Este cambio supuso modificar la parte de la aplicación encargada de la gestión de los LEDs así como diseñar 2 nuevos cables:

- El cable que conecta el puerto SCSI II de 50 pines con el receptor: se modificó de forma que uno de los terminales volvía a entrar dentro del Core, o más concretamente, hasta un nuevo puerto DB9 que se añadió al chassis.
- EL cable que conecta el nuevo conector DB9 con la placa en la que se encuentran los LEDs.

3.2.2.2 Puertos serie

Para poder conectar los conectores GPIO y serie presentes en el módulo RTM de la placa VM6250 fue necesario hacer estos conectores accesibles desde el exterior de CC. Para

ello, se encargó el mecanizado de una carátula extraíble ubicada en la parte trasera del CC para poder alojar estos conectores DB-9. Además de esto, se diseñaron los cables que unían estos conectores adicionales con los puertos del RTM. Los 3 conectores DB-9 adicionales, así como el cableado se aprecian en la imagen inferior:

Figura 12 - Puertos serie adicionales

4. MIGRACIÓN DE LA BSP

Como se explicó en el plan de trabajo, la migración de la BSP se subcontrató a otra empresa (que llamaremos SUBC) con amplia experiencia en sistemas operativos de LynxOS. De hecho, el fundador de la empresa ocupó en el pasado el puesto de jefe de desarrollo de LynxOS.

La metodología de trabajo de SUBC está basada en el prototipaje. SUBC fue entregando sucesivas versiones de su implementación hasta llegar a la implementación final.

SUBC comenzó con la implementación necesaria para poder arrancar el sistema y tener una versión muy inicial del sistema operativo. Esta entrega permitirá a CORP hacer las primeras validaciones. A posteriori SUBC fue implementando las funcionalidades más críticas para el sistema, como son el ramkdisk, la consola, drivers de los puertos serie, driver del disco duro y de otros dispositivos menos críticos como el CPLD o el PBIT.

Esta estrategia permitió poder hacer la migración de la aplicación el paralelo a la migración de la BSP. Además, esta metodología permite detectar los posibles bugs más rápidamente posible, ya que en vez de detectarse una vez esté el Sistema Operativo implementado, se fueron descubriendo en las sucesivas entregas.

Como ya hemos indicado en las secciones anteriores, para la migración del sistema operativo, se ha usado el Porting Kit de LynxOS 3.1.0a, que requirió rellenar una serie de funciones vacías en el código. La estrategia para llevar a cabo la implementación fue la siguiente:

- Recoger toda la información necesaria del hardware necesaria. Esta es la parte más importante de todo el proyecto, ya que la migración de la BSP se hace en función a estas especificaciones. Este paso está ampliamente descrito en los apartados anteriores. Tarea llevada a cabo por CORP.

- Diseño de los componentes, definir qué drivers se necesitarán y qué interfaz tendrán. Esta tarea también se ha definido en secciones anteriores. Tarea llevada a cabo por CORP.

- Diseñar tests usando la herramienta Automatic Test Suite para ampliar la cobertura de los tests incluidos. El Porting Kit incluye un paquete llamado ATS (Automatic Test Suite). Éste paquete software contiene una serie de test automatizados. Estas pruebas, se pueden usar para que el cliente verifique que su implementación de la BSP es correcta y no tiene más problemas. A parte de los tests incluidos, esta herramienta permite extenderlos y crear nuevos test. Tarea llevada a cabo por CORP/SUBC.

- Implementar las funciones definidas el el Porting Kit, así como los drivers necesarios. Tarea llevada a cabo por SUBC.

- Usar los ATS para validar la implementación. Tarea llevada a cabo por CORP/SUBC.

Durante la creación de las sucesivas versiones de la BSP (la versión definitiva ha sido la 7.0) surgieron numerosos problemas que se ha ido resolviendo en las nuevas entregas. Algunos problemas más importantes detectados y resueltos fueron:

- Integración del PBIT (Power-up Built In Test) de la VM6250 con la BSP. Por un problema de diseño del PBIT, los puertos serie de la placa quedaban desactivados.

- Acceso a los 8 puertos de la placa TEWS RS-422. Por un problema en el mapeo de memoria de este driver, no era accesible acceder a algunos de los puertos.

5. ADAPTACIÓN DEL SW & AIV

5.1 PROCESO DE MIGRACIÓN DE LA APLICACIÓN

La aplicación está diseñada para ser lo más independiente del hardware posible. De esta manera, la portabilidad a otra plataforma compatible con la interfaz POSIX. Todo y así, la nueva plataforma hardware dispone de dispositivos con una interfaz distinta a la antigua, así que se necesitarán implementar un mínimo de cambios en el software.

En concreto, los cambios que realizaremos serán los relacionados con los siguientes dispositivos:

- Watchdog: Este dispositivo es completamente incompatible con la anterior versión de hardware. Así que se tendrán que reescribir las funciones que lo gestionan. En el anterior dispositivo el Watchdog podía tener cualquier valor, pero en el nuevo solo puede tener los siguientes valores: 250ms, 500ms, 1seg, 2seg, 4seg, 16seg y 32seg, 64seg, así que los tiempos de las distintas tareas también se tendrá que adaptar a este nuevo comportamiento.

- PBit: Este dispositivo sirve para leer el resultado de los Power-up Built in Test que se ejecutan cuando se inicia la máquina. Además de que la plana VM6250 permite ejecutar más tests que la VMPC6a, este dispositivo tampoco tiene la misma interfaz, así que se deberán rehacer las funciones que lo usan. Además, daremos visibilidad a los nuevos tests disponibles, por lo que se requerirán cambios en las interfaces que reportan los tests a otros elementos.

- Tews: Este dispositivo no ha cambiado, de hecho es el mismo hardware que anteriormente. Todo y así, la tarjeta Tews dispone de 8 puertos 422, de los cuales nuestra aplicación solo usa 3. Se ha decidido usar 3 más para controlar los LEDs del frontal del chasis. Esta parte se describirá más en detalle en la siguiente PAC, donde describiremos las actividades a AIV (Assembly, Integration & Validation)

Como se explicó en la sección anterior, la migración de al aplicación se ha hecho en paralelo a la migración de la BSP, a medida que recibíamos versiones de la BSP permitiendo el uso de más dispositivos, se podían modificar y ejecutar más funcionalidades de la aplicación.

Las primeras versiones de la aplicación no tenían ni driver SATA, ni Watchdog ni el PBIT implementados, por lo tanto, las partes del código que utilizaban Watchdog y el PBIT estaban deshabilitadas y la aplicación no arrancaba desde el disco duro sino desde la red (netboot).

A medida que se fueron recibiendo sucesivas versiones de la BSP, se fueron habilitando y probando todas las partes de la aplicación hasta dar por concluida la migración.

Los siguientes pasos inmediatos del proyecto consisten ejecutar los Unit Tests de la aplicación, los tests funcionales de la aplicación y posteriormente realizar la integración con el resto de elementos hardware de la estación. Todo esto se explicará con detalle en la siguiente PAC.

5.2 ESTRATEGIA DE TEST

A la hora de definir la estrategia a seguir para la campaña de test, debemos de definir entre es testeo de la BSP y el testeo de la aplicación migrada.

La estrategia de test que se ha establecido para la aplicación viene determinada en gran medida por la metodología que se establece en el estándar DO-178 para el software crítico DAL C. La estrategia de test se divide en 3 fases bien diferenciadas que se explicarán detalladamente en las secciones siguientes: Unit Testing, Validación a nivel de Core Computer y Validación a nivel de estación.

Se seguirá una metodología waterfall en la que se ejecutará cada una de las fases anteriormente especificadas de forma consecutiva. En caso de que cualquiera de las fases no se complete correctamente (algún test falla), se modificará el código de la aplicación y se comenzará de nuevo de acuerdo con el siguiente esquema:

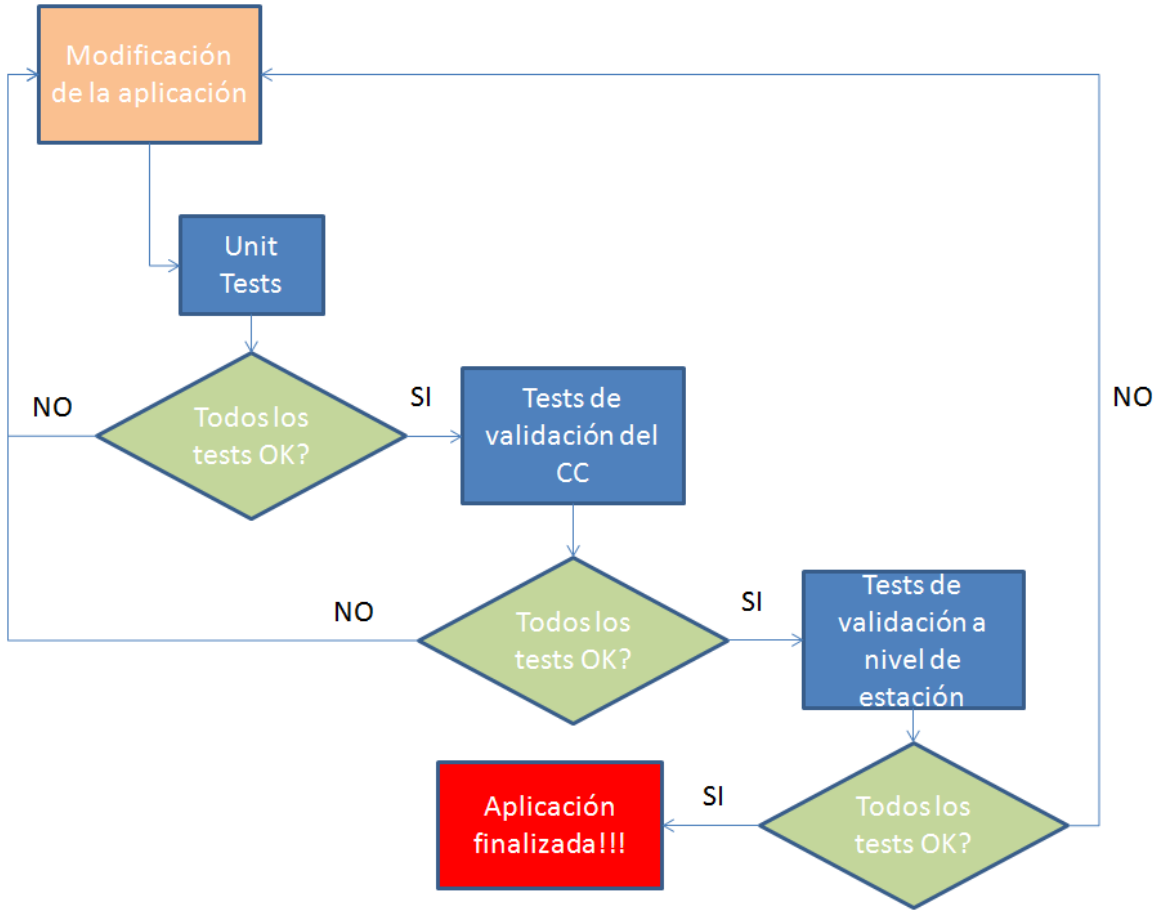


Figura 13 - Estrategia de test

Cada una de las fases se ejecuta completamente aunque esto no sería necesario en el caso de los unit tests, ya que no estamos probando ninguna funcionalidad de la aplicación sino cada función individualmente. Sin embargo, ya que la aplicación utilizada para el unit testing nos permite ejecutar todos los tests automáticamente en menos de media hora, lo ejecutamos completamente cada vez que hacemos un cambio en la aplicación.

Ejecutar los tests completos es algo obligatorio en caso de la validación de las funcionalidades de la aplicación, ya que la mayoría de las veces no poder predecir en qué forma, un pequeño cambio en una función puede tener en el resto de la aplicación, sobre todo en nuestro caso, en el que tenemos varios procesos corriendo concurrentemente.

Las siguientes secciones detallan cada una de las fases de la campaña de test.

5.2.1 Validación de la BSP

Tal y como se explicó en la PAC2, la BSP proporción las funcionalidades básicas del kernel que dependen del hardware de la placa.. La BSP se compone principalmente de un conjunto de módulos o componentes que implementan las interfaces hardware que son llamadas desde el sistema operativo cuando es necesario realizar operaciones hardware de bajo nivel y es responsable del start-up del sistema y de la configuración de los dispositivos integrados en la placa base. Además de esto, la otra parte principal de la BSP son los controladores de los dispositivos de la VM6250

Junto con el Porting Kit, LynuxWorks proporciona el ATS (Automatic Test Suite) que contiene numerosas test suites. Esta herramienta puede ser utilizada por los desarrolladores para escribir y ejecutar tests para validar la nueva BSP.

Los tests se clasifican en las siguientes categorías:

- BSP tests: Verifican las nuevas funciones implementadas para la BSP que han sido expresamente desarrolladas para este proyecto y que ya se definieron en profundidad en la PAC2.

- Driver tests: Verifican los nuevos driver desarrollados en el marco de este proyecto y que se describieron en la PAC2.

- Non Regression Tests: Conjunto de tests proporcionado por KynuxWorks y que constituye el subset mínimo de tests que cualquier BSP debe superar.

Los tests ejecutados fueron los siguientes:

Categoría	Test
BSP test	Memory allocation test
BSP test	Interrupt Controller test
BSP test	Serial Console test
BSP test	PCI BUS test
BSP test	Real Time Clock test
BSP test	Board Reset test
BSP test	I2C test

Driver test	Serial Port Driver test
Driver test	Ethernet Interface Driver test
Driver test	SATA Hard Disk Driver test
Driver test	CPLD (Watchodg & GPIO) Driver test
Driver test	PMC866 RS422 Ports Driver test
Non Regression test	Aburto performance test
Non Regression test	Application writer test
Non Regression test	Bash Test
Non Regression test	Byte Benchmark performance test
Non Regression test	Dhrystone Performance Test
Non Regression test	Library Function Test
Non Regression test	Memory Allocation Test Suite
Non Regression test	POSIX.1b Test
Non Regression test	POSIX 1b Virtual Memory Test

Tabla 10 – Tests de validación de la BSP

En nuestro caso, los tests fueron escritos por la empresa subcontratada encargada de realizar la migración de la BSP y nuestro trabajo consistió en ejecutar los tests sobre nuestra plataforma verificar el buen funcionamiento o reportar los posibles problemas encontrados para su posterior solución.

5.2.2 Unit testing

Una vez finalizada la adaptación del código fuente de la aplicación, llega el momento de hacer el Unit Testing. En nuestro caso, el Unit Testing se ha realizado utilizando la herramienta de IBM Rational Test Real Time.

Mediante Unit Testing, determinamos si cada una de las funciones implementadas en el código cumplen con sus especificaciones. Idealmente, cada test case es independiente de los demás, es decir, probamos cada función independientemente del resto de la aplicación. Para poder probar cada función independientemente, sustituimos las llamadas a otras funciones por stubs. Un stub es una función que nos devuelve el resultado que devolvería la función a la que sustituye sin tener que ejecutarla. De este modo, como el stub siempre devuelve un valor correcto, en caso de que el test falle, sabemos que el fallo se encuentra en la función que está siendo testada y no en una de las funciones a las que se llama desde dicha función.

Si la función que implementa el stub puede retornar varios valores (por ejemplo OK y NOK), haremos tantos tests sean necesarios para probar todas las configuraciones posibles.

Si durante la ejecución de los unit tests, o de cualquiera de los tests posteriores (test a nivel del CC o a nivel de estación) detectamos un problema y hay que modificar el código, habrá que modificar los unit tests correspondientes a las porciones de código que cambie y posteriormente habrá que ejecutar las campañas de test a nivel de CC y de estación completas según el esquema descrito en la figura 4-1.

Para nuestro caso, la realización de los unit tests consistió en las siguientes taréas:

- Configuración de la herramienta de test (IBM Rational Test Real Time) para funcionar sobre un nuevo target (placa VM6250). Esto es debido a que durante la campaña de unit test, el código que se testea correo sobre el hardware real.

- Modificación los tests correspondientes a los ficheros de código fuente modificados: los tests encargados de validar las funciones modificadas tuvieron que ser modificador para adaptarse a los cambios realizados.

- Ejecución de la campaña completa de unit test: una vez modificados los tests necesarios, se ejecutan todos los tests para obtener el report final que genera la aplicación. Esto, además de los resultados de todos los tests, nos permite obtener determinadas métricas, como el coverage o la complejidad ciclomática que deben de ser compatibles con las restricciones descritas en el documento DO-178 para el código DAL-C.

5.2.3 Validación de la aplicación a nivel de CC

Para validar la aplicación a nivel de Core Computer, es necesario simular el entorno real de funcionamiento pero sin utilizar los componentes hardware reales a los que el CC se conectar en el entorno real. Para esta tarea utilizamos un simulador del receptor GPS (Receiver simulator) y otro simulador que simula el módulo de monitorización del reloj atómico. Esto es debido a que debemos tener completamente controlados todos los inputs que la aplicación recibe de forma que en caso de encontrar un problema, estemos completamente seguros de que el problema reside en la aplicación y no en el resto del hardware de la estación.

Además de esto, debemos ser capaces que simular 3 tipos de eventos en el hardware conectado:

- Simulación del estado nominal de los equipos: los simuladores indican un funcionamiento nominal de los dispositivos para poder simular el funcionamiento de la aplicación en estado estacionario.

- Problemas hardware: tanto el receptor como el reloj atómico disponen de infinidad de alarmas para indicar diversos problemas dentro de estos dispositivos ante los que el Core Computer debe responder de forma diferente en cada uno de los casos. Por ejemplo, el reloj atómico puede tener un problema (pérdida de potencia de salida en uno de los 5 puertos o drift respecto a la señal de 10MHz) o el Receptor puede generar alarmas indicando diversos problemas como en la fuente de alimentación, corrupción de la memoria flash donde se almacena el ejecutable…

- Control de SIS (Signal In Space): el receptor proporciona al Core Computer mensajes con la información obtenida de los satélites a través de la antena. En la operación normal, podemos tener un diferente número de satélites en función del estado orbital, podemos tener jammer o multipath debido a interferencias o reflexiones de la señal… y en todo momento debemos de ser capaces de controlar estas situaciones (incontrolables con un receptor real) para ver si el CC responde correctamente a todos los posibles estímulos.

La figura siguiente representa de forma esquemática el entorno de test:

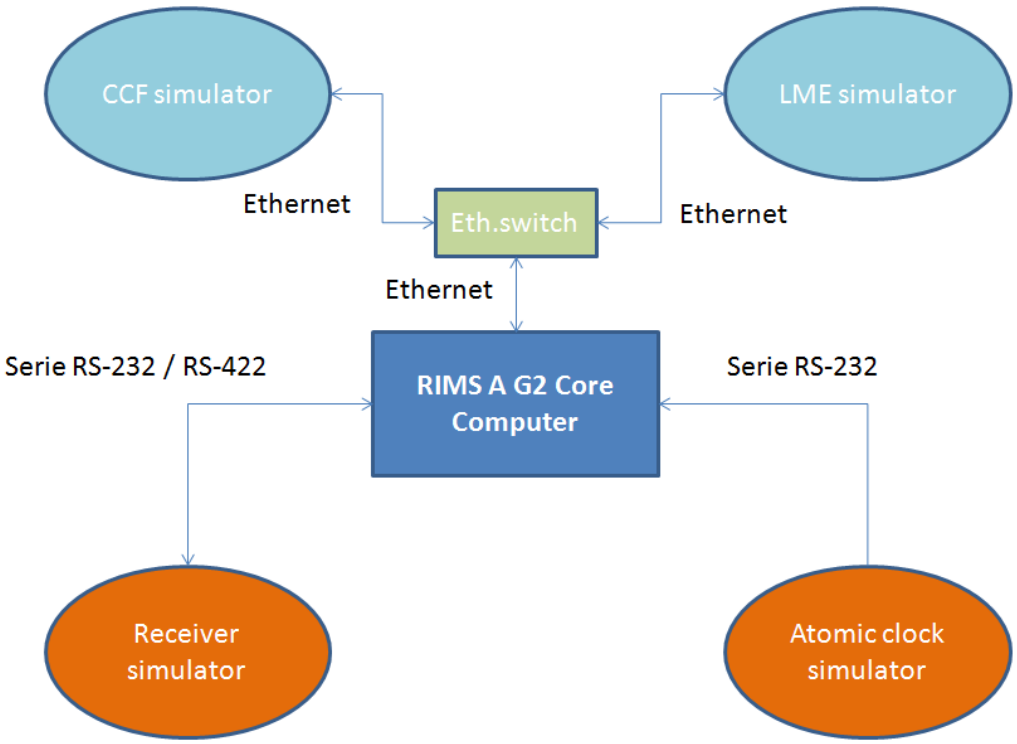


Figura 14 - Entorno de validación del Core Computer

En la figura anterior aparecen otro 2 simuladores de los que no hemos hablado:

- CCF simulator: Simula el Central Control Facility, centro de control desde el que se controlan las estaciones RIMS instaladas. A través de este simulador, podemos enviar cualquier comando a la estación o realizar cualquier otra operación necesaria como instalar ficheros de SW o configuración.

- LME simulator: Simula el Local Maintenance Equipment, equipo que un operador puede llevar consigo y conectar directamente a una estación en caso de haber detectado un problema para actividades de mantenimiento. Tiene unas funcionalidades similares a las del CCF simulator.

Cuando generamos una nueva versión de SW para la RIMS A, no se ejecuta toda la batería de test completa (se tarde unas 3 semanas en ejecutar y verificar todos los tests) sino que se ejecutan los test correspondientes a las nuevas funcionalidades o funcionalidades

modificadas y un subconjunto de test mínimo que prueba las funcionalidades básicas y más importantes de la estación.

Sin embargo, en este caso, al haber modificado la BSP y utilizar un nuevo hardware, se decidió ejecutar la batería completa de test.

Tras ejecutar la campaña completa de test solamente se encontró un problema: la aplicación no detectaba la desconexión del CCF o del LME simulator. Este problema se solución editando un fichero de configuración /sys/devices/hbtcpip_info.c, y más concretamente los parámetros tcp_keepidle y tcp_keepintvl que modifican los timeouts del "keep alive". En TCP/IP se utiliza los paquetes keepalive para detectar si un determinado host está caído. Es totalmente configurable, de manera que podemos determinar cada cuantos segundos se envía esta señal o si inhibimos su uso.

Pase a esta modificación, no fue necesario modificar el Unit Test ni repetir la campaña completa de tests ya que no se modificó ni una línea de código fuente. Fue suficiente con repetir los 3 tests de validación involucrados.

5.2.4 Validación de la aplicación a nivel de estación

Una vez que el Core Computer está validado, llega el momento de probar la estación completa. A nivel de estación se realizan 3 tipos de tests:

- Tests de fabricación: dentro de este grupo estarían los tests que verifican el consumo de potencia de la estación, el calentamiento o el ruido emitido. Todos estos parámetros están limitados por unos valores descritos en los requisitos.

- Tests de validación: este subgrupo de tests validan las funcionalidades de la estación completa. Al contrario que cuando se validó el Core Computer, en este caso, se utiliza la configuración definitiva con los equipos reales.

- Tests de Witnessing: cuando llega el momento de entregar la primera estación al cliente (ESA), estos se desplazan hasta nuestra empresa y se ejecuta una subgrupo de tests de validación que son lo suficientemente significativos como para que el cliente puede realizar la aceptación de la estación.

Tanto los tests de validación como los de witnessing, se realizan con la siguiente configuración hardware:

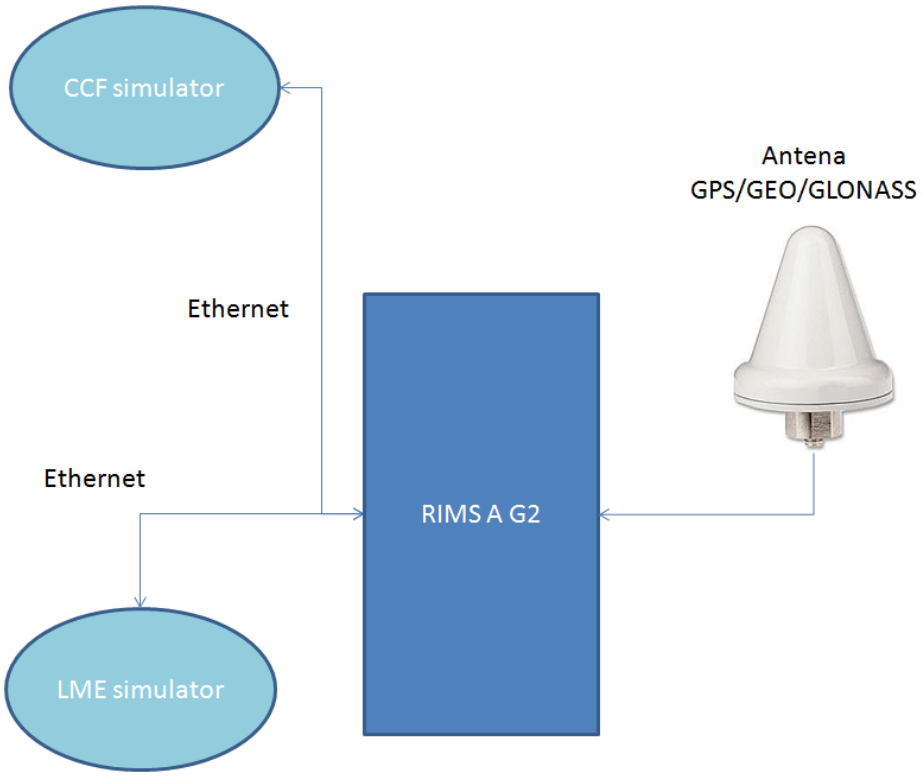


Figura 15 - Entorno de validación de la estación

En el momento de escribir esta PAC, aún no se ha comenzado con la validación a nivel de estación. Hemos acabado los tests a nivel de Core Computer satisfactoriamente y la validación a nivel de estación está planeada para que comience a mediados de junio.

6. MARCADO CE

Una vez finalizadas las actividades AIV, y en paralelo con las actividades migración de la aplicación, se realizaron las actividades de marcado CE.

Por requerimiento del cliente, la RIMS A G2 debe tener el marcado CE para su comercialización en Europa. Si bien las estaciones también se instalan en países no europeos como Canadá, Sudáfrica, Egipto… los emplazamientos en los que se instalan, se consideran territorio europeo por lo que no es necesario cumplir ninguna otra regulación. Además del marcado CE de la estación completa, el cliente también solicitó el marcado CE del Core Computer individualmente.

Las actividades de marcado CE deben de ser realizadas por un laboratorio certificador. En este caso, las actividades fueron realizadas en APPLUS + LGAI Certification Technological Center (laboratorio acreditado por Aenor para la certificación de equipos bajo la norma CE entre otras). Estuvimos presentes durante la realización de las pruebas para operar tanto el CC como la estación completa para verificar que el funcionamiento durante el proceso de test correspondía con el comportamiento nominal de los equipos.

Tanto la estación como el Core Computer pasaron todos los tests sin problemas. Antes de la ejecución de las pruebas, no se esperaban mayores problemas ya que por un lado la plataforma de estación y la mayor parte de los equipos estaba basada en la plataforma anterior que estaba certificada y consolidada. Por otra parte, los componentes del Core Computer son componentes comerciales que tienen marcado CE individualmente, por lo que no se esperaban problemas para obtener el marcado CE para el conjunto, aunque en muchos casos, sabemos por experiencia que CE + CE no siempre es igual a CE.

6.1 NORMATIVA APLICABLE

Los estándares que son aplicables tanto a la estación como al Core Computer son los que aplican a un equipo de tecnologías de la información en un entorno industrial:

- Compatibilidad electromagnética
    - UNE-EN 55022:2008 + A1:2008: Equipos de tecnología de la información. Características de las perturbaciones radioeléctricas. Límites y métodos de medida.
    - UNE-EN 55024/A2:2004: Equipos de tecnología de la información. Características de inmunidad. Límites y métodos de medida.
    - UNE-EN 61000-3-2:2006: Compatibilidad electromagnética (CEM). Parte 3-2: Límites para las emisiones de corriente armónica de entrada <=16 A por fase.
    - UNE-EN 61000-3-3:2009: Compatibilidad electromagnética (CEM). Parte 3-3: Límites. Limitación de las variaciones de tensión, fluctuaciones de tensión y

flicker en las redes públicas de suministro de baja tensión para equipos con corriente asignada <= 16 A por fase y no sujetos a una conexión condicional.

- Seguridad eléctrica

    - UNE-EN 60950-1:2007 + CORR 2007 + A11:2009: requerimientos esenciales bajo las especificaciones aplicables de los estándares UNE-EN 60950-1: 2003 + CORR: 2004 + A11: 2004.

6.2 ESTRATEGIA DE CERTIFICACIÓN

A la hora de hacer el marcado CE de un equipo, los tests se deben ejecutar de forma que el equipo se encuentre en unas condiciones lo más parecidas posibles al funcionamiento nominal. Por ejemplo, en el caso de certificar un PC, no es necesario que los tests se hagan mientras corre un software específico sino que lo importante es que mientras se ejecutan los tests, se estén ejercitando todos los interfaces así como que estén activos todos los dispositivos interno. De esta forma, estaremos simulando las peores condiciones posibles de funcionamiento.

Para certificar el Core Computer, se diseñó una aplicación ad hoc que se dedicaba a enviar y recibir paquetes de datos por cada uno de los interfaces (se cortocircuitaron las entradas y las salidas) mientras que se escribían los resultados en el disco duro. Al mismo tiempo, estaba establecida una comunicación TCP/IP con un equipos auxiliar al que se le envía el estatus de cada uno de los interfaces. De esta forma, somos capaces de saber el estado del Core Computer durante los tests. Si una vez terminado el test, no se ha detectado ningún problema en el equipo, el test se considera superado.

Para la certificación de la estación completa se utilizó la primera versión de la aplicación migrada con funcionalidades limitadas. Lo único que era necesario es que la estación pudiera permanecer en uno de los modos sin transiciones y sin irse al modo Fail.

7. ESTRATEGIA DE CUALIFICACIÓN

El software que corre en el Core Computer es por requisitos del cliente, software crítico de nivel C (DAL-C). El documento *"DO-178, Considerations in Airborne Systems and Equipment Certification*" define y especifica los distintos niveles de criticidad o DAL (Design Assurance Level) para software embarcado en funcion de los efectos que pueden provocar un fallo en el software. Estos niveles son los siguientes:

- A: Catastrófico: Puede provocar que una aeronave se estrelle.
- B: Peligroso: Se reduce la maniobrabilidad, puede que los pasajeros resulten heridos e incluso muertos.
- C: Importante: Reduce la seguridad de la aeronave, provoca un mayor trabajo para la tripulación y molestias para los pasajeros.
- D: Menor: El fallo es apreciable pero no disminuye la seguridad de la tripulación.
- E: Sin efecto: El fallo pasa desapercibido.

El que el software sea DAL-C incluye una serie de restricciones importantes y obliga a tener una exhaustiva campaña de test, así como a la certificación de los COTS (en este caso el sistema operativo) que se utilice.

Dado que LynxOS 3.1 no es certificable, en el pasado se certificó el software mediante PSH (Product Service History), estrategia que explicaremos detalladamente más adelante.

Para este proyecto, partimos de la premisa de que el porcentaje de líneas de código modificado es menos del 20%, por lo que según el DO-178, el impacto del cambio puede considerarse menos.

El plan propuesto para la certificación el nuevo sistema, y que puede ser usado para futuros procesos de certificación dentro de EGNOS cubre los siguientes aspectos:

- Capa de aplicación: La aplicación que se ha modificado estaba desarrollada y cualificada de acuerdo con los requisitos DAL C. Dado que como ya se ha explicado, los cambios en esta capa se consideran menores (menos del 20%), no es necesario desde el punto de vista formal definido en DO-178, volver a justificar la cualificación de la aplicación.
- Capa del sistema operativo: dentro del marco de este proyecto, dado que la aplicación de ha migrado a una nueva placa, la cualificación de los elementos que forman la capa del sistema operativo (sistema operativo y API), es necesaria para mantener el mismo nivel de cualificación que en el pasado. Las piezas que se consideran como partes de esta capa son: el sistema operativo, el stack TCP/IP, la BSP y las librerías. Para cualificar estos elementos, se realizarán las siguientes 2 actividades:
    - API COTS Testing: La API de sistema operativo será analizada por una empresa externa para confirmar que es los suficientemente estable y robusto como para considerar que la contribución a la pérdida de continuidad del sistema es entendida y acotada. Este análisis es realizado mediante una

campaña de test específica para LynxOS 3.1. Por ejemplo, se verifican cosas como que el valor de la variable errno sea el correcto antes diferentes situaciones de fallo de las llamadas al sistema.

- Product Service History (PSH) sobre la RIMS A G2 en un entorno representativo durante 1 año. El PSH consiste en mantener el sistema en un entorno representativo del entorno operacional real. Para ello debemos intentar reproducir las operaciones que el operador real del sistema (European Satellite Service Provider) en cuanto a configuración del sistema, descarga de logs, monitorización de parámetros, ciclos de reset, análisis de las posibles averías HW o fallos SW… de forma que una vez pasado el año, tengamos una cantidad suficiente de información como para determinar que la aplicación es estable y que la contribución de la estación a la pérdida de continuidad del sistema es cuantificables y acotada.

Una vez realizadas estas actividades, se considerara que la RIMS A G2, o más concretamente, su software está cualificado como DAL C de acuerdo con el estándar DO-178. Tal y como se especificó en el diagrama de Gantt incluido en la PAC1, la finalización de estas actividades coincide con la finalización del proyecto y el comienzo de las actividades de mantenimiento y garantía.

8. TAREAS DESARROLLADAS DURANTE LA EJECUCIÓN DEL PROYECTO

En el proyecto que se describe en este TFC intervenimos 5 personas: un Project manager, un Technical manager y 3 ingenieros de desarrollo entre los que me encuentro.

De los 3 desarrolladores, yo soy el que tiene un perfil más hardware y el que más tiempo lleva trabajando con la aplicación (sobre la antigua plataforma), mientras que mis compañeros tienen un perfil más de programador.

Debido a esto, de las actividades descritas en la presenta en esta memoria, he realizado las siguientes tareas:

- Responsable del análisis a nivel hardware de las dos plataformas (VMPC6a y VM6250)

- Responsable del diseño y actividades de AIV tanto del Core Computer como de la estación completa

- Responsable de las actividades de marcado CE realizadas en el laboratorio certificador

- Líder de la campaña de test de la aplicación a nivel de Core Computer

- Líder de la campaña de test de la aplicación a nivel de estación (actividad aún no realizada)

- Operador de la estación durante la fase de PSH (actividad aún no realizada)

9. CONCLUSIONES

Se ha explicado el proceso de migración de una aplicación embebida perteneciente a uno de los subsistemas del segmento de tierra del sistema de aumentación satélite europeo.

Se ha explicado el análisis realizado tanto a nivel hardware de las dos plataformas como a nivel software de la aplicación a migrar y sobre todo de la BSP.

Se ha explicado el proceso de migración de la aplicación y de la BSP y la estrategia de test realizada para validar la aplicación.

Además, se ha explicado otra problemática relacionada con la actividad realizada como es el marcado CE de la nueva plataforma hardware y la estrategia de cualificación a realizar, introduciendo la problemática relacionada con el software crítico.

Además se ha enumerado las tareas en la que he participado activamente y aquellas de las que he sido responsable.

En estos momentos, se está a punto de validar la estación completa con todos los equipos integrados.

Personalmente estoy disfrutando mucho con este proyecto ya que es la primera vez que participo en un desarrollo, ya que hasta éste momento mi trabajo se ha centrado fundamentalmente en actividades de mantenimiento HW/SW pero sin generar nuevas versiones de software.

FIN DEL DOCUMENTO