EM4000 Software Architecture

|  |  |
| --- | --- |
| Core Team Review/Approval | |
| Core Team Member | Name |
| Core Team Manager | L. Lesenne |
| Development Engineering |  |
| Software Project Leader | F. Pasquier |
| Hardware Project Leader | C. Lejus |
| Verification & Validation Engineering |  |
| Quality Management |  |

|  |
| --- |
| Subject : **EM4000 – Software Architecture**  This document aims at describing the software architecture of EM400x HD encoder. |

|  |  |  |  |
| --- | --- | --- | --- |
| REVISION HISTORY | | | |
| Revision | Description | Date | Author(s) |
| 0.1 | 1st draft | 20/09/2010 | J. BARDON – J.L. LEBLANC |
| 0.2 | Compléments (QAP express, ASI, IP IN) | 27/09/2010 | J. BARDON – J.L. LEBLANC |
| 0.3 | Transferts de flux | 06/10/2010 | J. BARDON – J.L. LEBLANC |
| 1.0 | Mise à jour | 25/04/2012 | J.BARDON |
| 1.1 | Mise à jour | 26/06/2013 | J. BARDON |

Table of Contents

[1 Scope 5](#_Toc360008452)

[1.1 Subject and Identification 5](#_Toc360008453)

[1.2 Product overview 5](#_Toc360008454)

[1.3 Document overview 5](#_Toc360008455)

[2 Referenced Documents 6](#_Toc360008456)

[3 Application software - framework EM4000 7](#_Toc360008457)

[3.1 Environnement de l’A.S. 7](#_Toc360008458)

[3.2 Architecture logicielle de l’A.S. 7](#_Toc360008459)

[3.3 Interfaces Externes 8](#_Toc360008460)

[3.3.1 Interface http 8](#_Toc360008461)

[3.3.2 Interface SNMP 8](#_Toc360008462)

[3.3.3 Interface NMS 9](#_Toc360008463)

[3.3.4 Interface LCD 9](#_Toc360008464)

[3.3.5 Interface Console 9](#_Toc360008465)

[3.4 Interfaces de test 9](#_Toc360008466)

[3.5 Control & Command Middleware 9](#_Toc360008467)

[3.5.1 Config Manager 9](#_Toc360008468)

[3.5.2 Status Manager 10](#_Toc360008469)

[3.5.3 Board Manager 10](#_Toc360008470)

[3.5.4 Ident Manager 10](#_Toc360008471)

[3.5.5 Monitor 10](#_Toc360008472)

[3.5.6 ENCCTL 10](#_Toc360008473)

[3.6 Installation 10](#_Toc360008474)

[3.7 Contrôle des process 11](#_Toc360008475)

[3.7.1 Boot 11](#_Toc360008476)

[3.8 Contrôle des ressources 11](#_Toc360008477)

[3.8.1 Process Resource Controller 11](#_Toc360008478)

[3.8.2 Process IP Controller 13](#_Toc360008479)

[3.8.3 Process orientés stream 13](#_Toc360008480)

[3.9 Transferts de flux 14](#_Toc360008481)

[3.9.1 Hardware -> software 15](#_Toc360008482)

[3.9.2 Software -> software 15](#_Toc360008483)

[3.10 Interfaces avec le hardware 15](#_Toc360008484)

[3.10.1 Interface Pegasus 15](#_Toc360008485)

[3.10.2 Interface carte 16](#_Toc360008486)

[4 Fonctions Système 16](#_Toc360008487)

[5 Distribution / OS 16](#_Toc360008488)

[6 Glossaire 16](#_Toc360008489)

# Scope

## Subject and Identification

.

## Product overview

See […](#_Referenced_Documents)

## Document overview

# Referenced Documents

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ref.** | **Identification** | **Title** | **Issue Date** | **Origin** |
| [01] | [http://thomson-networks.thomnet.com/km/HeadEnd/Engineering/Andromede/Project\_documents/Forms/AllItems.aspx?DocumentID=245](https://sharepoint.harmonicinc.com/km/headend/Engineering/Andromede/Project_documents/Forms/AllItems.aspx?DocumentID=245) | Hard real-time platform solution | 20/08/2010 | J. DANIEL |
| [02] | [http://thomson-networks.thomnet.com/km/HeadEnd/Engineering/Andromede/Project\_documents/Forms/AllItems.aspx?DocumentID=279](https://sharepoint.harmonicinc.com/km/headend/Engineering/Andromede/Project_documents/Forms/AllItems.aspx?DocumentID=279) | Spécifications Promain 4.1 | 21/05/2001 | P. GRAVOILLE – B. LE VERN |
| [03] | [http://thomson-networks.thomnet.com/km/headend/Engineering/Andromede/Project\_documents/Forms/AllItems.aspx?DocumentID=2230](https://sharepoint.harmonicinc.com/km/headend/Engineering/Andromede/Project_documents/Forms/AllItems.aspx?DocumentID=2230) | Andromede Filesystems | 07/06/2011 | J. BARDON |
| [04] | [http://thomson-networks.thomnet.com/km/headend/Engineering/Andromede/Project\_documents/Forms/AllItems.aspx?DocumentID=2157](https://sharepoint.harmonicinc.com/km/headend/Engineering/Andromede/Project_documents/Forms/AllItems.aspx?DocumentID=2157) | SDD Bellerophon | 19/05/2011 | B. GUERIN |
| [05] |  |  |  |  |
| [06] |  |  |  |  |

# Application software - framework EM4000

## Environnement de l’A.S.

Le schéma suivant présente l’environnement du framework métier / application software et les protocoles supportés sur les interfaces externes :



## Architecture logicielle de l’A.S.

Cette architecture est basée sur le framework des équipements ViBE et plus particulièrement de la plateforme Phoenix (encoder SD).

Cette architecture en découpée en 3 couches logicielles :

* Les process de la couche d’interface externe permettent d’adapter le protocole des interfaces externe dans le protocole interne NCCP et inversement (sauf MUXSTAT).
* Les process de la couche middleware permettent le management de l’équipement dans les différentes aires fonctionnelles (identification, configuration, performances) et la gestion des services MPEG générés et incidents
* Les process  de la couche resource Management permettent la gestion des interfaces matérielles via le kernel ou le co-kernel.

Voir **PROMAIN**. Voir **NCCP**.

**PROMAINPROMAIN**



## Interfaces Externes

### Interface http

Fournie un service WEB avec serveur http minimal basé sur EHS.

Fonctions :

* Authentification des utilisateurs.
* Configuration de l’équipement (hors Flextream).
* Sauvegarde et rappel de configurations.
* Téléchargement et sauvegarde de configurations externes.
* Affichage des alarmes.
* Monitoring des débits.
* Rebooter l’équipement.
* Informations hardware/software.

Le process **Webapp** est chargé du support du http.

### Interface SNMP

L’agent SNMP supporte le protocole SNMP v2.C.

Il utilise les services de la librairie Netsnmp.

MIB basée sur celle de ViBE/SD/HD.

Fonctions :

* Informations hardware/software.
* Rebooter l’équipement.
* Alarmes et évènements (traps).
* Gestion des destinataires de traps.
* Rappel des configurations prédéfinies.

Le process **Snmpapp** implémente l’agent SNMP.

### Interface NMS

Supporte le protocole de configuration MUXCONFIG depuis un XMS/XMU.

Fonctions :

* Configuration complète de l’équipement (plan de service).
* Configuration de fonctions système (Flextream).
* Récupération de la configuration.

L’agent Muxconfig est implémenté par le process **Muxconfig**.

### Interface LCD

L’interface LCD regroupe un module hardware composé d’un écran et d’un clavier et un module software sous la forme d’un process de contrôle **Lcdctrl**.

Permet de :

* Définir les adresses IP de l’équipement.
* Rebooter l’équipement.
* Visualiser les alarmes.
* Informations hardware/software.
* Rappeler des configurations sauvegardées.

### Interface Console

Connexion possible en SSH v2.

## Interfaces de test

Digitest 🡪 NCCP.

Console 🡪 ssh.

## Control & Command Middleware

### Config Manager

Supporte la configuration de l’équipement en provenance des interfaces externes. Route des éléments de configuration vers les bons process.

Ce process est en charge du routage de la configuration (au format NCCP) vers les process destinataires.

Ce process administre la configuration courante, les configurations prédéfinies et les sauvegardées.

*Interlocuteurs :*

**Entrée :** Tous les adaptateurs d’interfaces

**Sortie :** Tous les process

### Status Manager

Ce process est en charge du management des alarmes et des events log de l’équipement dans une database.

*Interlocuteurs :*

**Entrée :** Tous les process

**Sortie :** Tous les agent d’interface externe

### Board Manager

Ce process est charge de la gestion de la topologie interne du codeur.

*Interlocuteurs :*

**Entrée :** Tous les process

**Sortie :** Tous les agents d’interfaces externes.

### Ident Manager

Ce process est en charge de l’inventaire des ressources HW/SW, packages logiciel de l’équipement.

*Interlocuteurs :*

**Entrée :** Tous les adaptateurs d’interfaces.

**Sortie :** Tous les process .

### Monitor

Ce process est en charge de maintenir une base de données des informations relatives aux débits (TS, services, composantes) des flux TS générés et incidents.

*Interlocuteurs :*

**Entrée :** ENCCTL.

**Sortie :** Tous les agents d’interfaces externes.

### ENCCTL

Chargé de l’application de la configuration du service plan (contrôle de la cohérence des paramètres, unicité des PIDs, vérification des débits des multiplex …)

Création des tables de signalisation à partir de la configuration appliquée (signalisation ISO (PAT, CAT, TSDT, PMT)+ signalisation DVB : SDTactual, EITp/f)

Configuration des codeurs, configuration du MUX.

Gestion du monitoring (représentation en TS/src/cmp avec débits associés aux composantes).

*Interlocuteurs :*

**Entrée :** CONFIMNG

**Sortie :** RCTL, MONITOR, STATUSMNG

I

## Installation

Le process **Loader** est chargé de la :

* Gestion des logiciels.
* Gestion des N banques/versions.

Le process **Vtbox** gère les options et certaines com NCCP inter-process.

## Contrôle des process

### Boot

Boot est le lanceur Promain qui forke tous les process. Les process possèdent 2 étapes d’initialisation avec point de synchro assuré par Boot. Toutes les allocations de mémoire doivent être faites dans la première phase d’initialisation (user init 1).

## Contrôle des ressources

La couche de contrôle des ressources supervise les ressources suivantes :

* Hardware : carte Pegasus et sous-périphériques.
* Software : process de traitement des flux.



### Process Resource Controller

Le Resource Controller regroupe les fonctions suivantes :

* Gestion de la topologie carte 🡪application type.
* Gestion de la topologie chip (Pegasus) à travers la PMAPI.
* Download Pegasus, FPGA et NIOS.
* Gestion de la configuration global et split vers chaque entité.
* Gestion et concentration des alarmes du hardware et des modules software.
* Gestion et concentration des statuts.

Le RC est un process unique dans le system software. Il est forké par le lanceur Promain (boot).

Il contrôle les N cartes présentes dans le chassis.

Il gère la topologie carte et chip en fonction du type d’équipement connu au boot de la machine et du choix de l’application type par board configurée par le client.

Ce process est responsable du cycle de vie des process de traitement (audio coders, video coders, schedulers).



*Interlocuteurs :*

**Entrées :**

* **Encctl** : Configurations de niveau encodeur, configuration du mux.

**Sorties :**

* **StatusManager** : Alarmes.
* **Encctl** : Monitoring (débits, afd, wss).

#### Cycle de vie des process

### Process IP Controller

**Ipctl** configure les 2 interfaces IP et encapsule (UDP et RTP/UDP) les flux TS générés par le mux.

*Interlocuteurs :*

**Entrée :**

* **ConfigManager** : Configuration des interfaces, configuration de l’encapsulation.

**Sortie :**

* **Socket IP**.

### Process orientés stream

Process Promain. Ces process sont forkés par RCTL qui contrôle leur état.

La topologie matérielle étant connue au boot, RCTL instanciera le nombre max de process necéssaires en fonction du type de produit (EM4002/4/6/8).

Au boot (user\_init\_n) chacun des process de traitement créé son environnement de communication.

Les process forkés par RCTL sont instanciés avec le dimensionnement maximum pour le produit.

Le main thread étant en attente bloquante sur les communications asynchrones de type Promain (sans réponse) tout traitement récurrent doit être géré dans un ou plusieurs threads lancés en fonction de la configuration.

Le process est en charge de la gestion de la conf et de son instant d’application.

Rctl interroge périodiquement (récurrence de 250 ms à 1s suivant les cas) les alarmes et les statuts (débits, standard détecté, etc) auprès des process de traitement qu’il contrôle. Les process gèrent leurs flags d’alarmes de manière complètement asynchrone du polling de RCTL.

*Interlocuteurs :*

**Entrée :**

* **RCTL** : Configuration.
* **Driver**: flux.

**Sortie :**

* **RCTL** : Alarmes, statuts, configuration.
* **Mux**: flux.

#### Mécanisme d’attachement

*(Voir* *Transfert de flux : Hardware -> software : )*

Lors de l’attachement initial d’un process à un pipe/heap ou lors du ré-attachement due à un plantage le process notifie à la tâche Xenomai son état en envoyant un message Ready dans le pipe. La tâche Xenomai doit alors flusher le pipe de commande (et éventuellement le buffer dans la heap) afin de reprendre un transfert normal.



#### Codeur Audio

Ces process codent l’audio PCM routée par les shcedulers. Il peuvent êtres des codeurs (DD, AAC, MP1L2) ou bien des décodeurs DolbyE. Dans le cas des décodeurs DolbyE ils alimentent alors des codeurs avec une partie du flux DolbyE décodé.

#### Codeur Video Low-Res

<TBD>

#### Schedulers Video

Proxy de transfert des paquets TS du bus PCI-e (carte Pegasus) vers l’interface d’entrée du mux ou bien les codeurs low-res.

#### Schedulers audio

Proxy de transfert des paquets TS du bus PCI-e (carte Pegasus) vers l’interface d’entrée des codeurs/décodeurs audio.

#### IP input process

Liste des traitements :

* Gestion des transferts avec l’entrée IP.
* FEC -1, dégiguage.
* Transfert vers la carte Pegasus du flux.

Le timestamping des paquets est conservé.

#### ASI Input process

<TBD>

#### ASI Output process

<TBD>

## Transferts de flux

### Hardware -> software

Principe de transfert de flux en entrée des process de traitement :



La tâche Xenomai produit du flux vers la *rt\_heap* et un flux de contrôle à travers des *rt\_pipe* vers chaque process de traitement.

La *rt\_heap* est partagée entre les process traitant le même type de flux.

Chaque process en attente bloquante sur son *rt\_pipe* lit les données relatives à son buffer d’entrée*.*

On utilisera une rt\_heap par type de flux (audio BB, Mpeg TS, video BB) et par carte : 4 cartes x 3 types de flux = 12 rt\_heap max. Et il existe autant de tâches Xenomai créées par le driver.

🡪**RCTL au fork des process fourni son attachement à la *rt\_heap* et à son *rt\_pipe*.**

### Software -> software

A travers une heap spécialisée créée par Chronos (SoftMRM) et gérée par RCTL. Charge à RCTL à travers son framework Mux de gérer les allocations et les connexions.

## Interfaces avec le hardware

### Interface Pegasus

Les configurations, alarmes et statuts d’une part et le download des chips d’autre part sont gérés en espace user grâce à la librairie *LibPegasusManager* (et le process Pmdaemon) linkée directement avec RCTL.

L’interface d’accès est PMAPI.

La topologie des chips Pegasus de la carte est contrôlée à travers la notion d**’application type** qui correspond à un certain nombre de gatewares (applications du fpga Pegavio). Ces Application types définissent la topologie complète des chips pour une carte. Ils sont configurables carte par carte à travers la PMAPI.

### Interface carte

Module Kernel Linux avec instanciation de tâche temps-réel Xenomai 🡪driver Bellerophon.

SDD Bellerophon : [http://thomson-networks.thomnet.com/km/headend/Engineering/Andromede/Project\_documents/Forms/AllItems.aspx?DocumentID=2157](https://sharepoint.harmonicinc.com/km/headend/Engineering/Andromede/Project_documents/Forms/AllItems.aspx?DocumentID=2157)

Bellerophon permet de configurer le gateware type du fpga Pegavio qui doit correspondre à une topologie de chips soit un Application Type. Charge à RCTL de synchroniser les différents changements d’états du fpga et des chips à travers la PMAPI.

# Fonctions Système

Flextream

# Distribution / OS

Le noyau linux (package ANKERNEL) est livré comme un module séparé. Il contient le scheduler temps-réel Xenomai.

Le noyau est un vanilla 3.4.32 avec Xenomai 2.6.2 + patch DMA32.

Le rootfs (package ANBASE) est un rootfs custom en initramfs basé sur Busybox.

Voir : [http://thomson-networks.thomnet.com/km/headend/Engineering/Andromede/Project\_documents/Forms/AllItems.aspx?DocumentID=2230](https://sharepoint.harmonicinc.com/km/headend/Engineering/Andromede/Project_documents/Forms/AllItems.aspx?DocumentID=2230)

# Glossaire

**PROMAIN** est un wrapper posix permettant une abstraction entre l’API et l’OS et offre les services suivants :

* Un mécanisme de synchronisation au lancement de l’application.
* Un mécanisme de communication interprocess.
* Un ensemble de tools de debug.
* Une encapsulation des sockets Unix.

Présentation Promain :

[http://thomson-networks.thomnet.com/km/headend/Engineering/Andromede/Project\_documents/Forms/AllItems.aspx?DocumentID=280](https://sharepoint.harmonicinc.com/km/headend/Engineering/Andromede/Project_documents/Forms/AllItems.aspx?DocumentID=280)

Spécifications Promain 4.1 :

[http://thomson-networks.thomnet.com/km/headend/Engineering/Andromede/Project\_documents/Forms/AllItems.aspx?DocumentID=279](https://sharepoint.harmonicinc.com/km/headend/Engineering/Andromede/Project_documents/Forms/AllItems.aspx?DocumentID=279)

**NCCP**(Nextream Control Command Protocol).

Le protocole NCCP définit les messages, les commandes et les données du protocole de C&C échangés entre les process de l’équipement.

Le data model NCCP est composé de descripteurs (les données du protocole) dans différentes classes (identification, configuration, supervision …).

Les messages NCCP sont échangés entre processs dans des communications promain.