

Uppgjord — <i>Prepared</i>	Faktaansvarig — <i>Subject responsible</i>	Nr — <i>No.</i>		
UAB/D/SF UABHAFE	UAB/D/SF			25/155 19-ANZ 212 08 Uen
Dokansv/Godk — <i>Doc respons/Approved</i>	Kontr — <i>Checked</i>	Datum — <i>Date</i>	Rev	File
UAB/D/RU (K. Isacson)	-	1999-11-16	A	

RP INTERPROCESSOR COMMUNICATION

Contents	Page
1 REVISION INFORMATION	2
2 INTRODUCTION	2
2.1 Application	2
2.2 Scope	2
2.3 Concepts	2
3 GENERAL SERVICE INFORMATION	4
3.1 Service Description	4
3.2 Service Exception Handling	4
3.3 Service System Recovery	5
3.4 Service Characteristics	5
3.4.1 Link handler names	5
3.4.2 Stand alone configuration	5
3.4.3 Restrictions	6
3.4.4 Characteristics	6
4 USE CASES	7
4.1 Search for a process	7
4.2 Attach a signal to a process	7
4.3 Detach a signal from a process	8
4.4 Notification of logical channel failure	8
5 SIGNAL IDENTIFICATIONS	8
6 TERMINOLOGY	9
7 REFERENCES	9

Nr — No.		
25/155 19-ANZ 212 08 Uen		
Datum — Date	Rev	File
1999-11-16	A	

1 REVISION INFORMATION

Table 1

Rev. A	First version of this document.	UABHAFE
--------	---------------------------------	---------

2 INTRODUCTION

2.1 Application

This interface description is valid for AXE regional processors of type RPP and RPI.

The RP operating system contains support for Inter Processor Communication between RPs. The service is based on the link handler concept and provides reliable and efficient communication between different RPs.

The link handler protocol stack uses Ethernet for communication between RPs. **Not** UDP/IP or TCP/IP.

Some of the following concepts and descriptions has been found in [1] since the function is based on the OSE Delta mechanism for distributed systems.

2.2 Scope

The scope of this document is to briefly describe the services provided by the operating system for an application designer. A more detailed description of the link handler concepts can be found in [1].

2.3 Concepts

Hunt An OSE Delta system mechanism, mainly used for multi-CPU communication, to get the process identity of a named process.

Hunt Path A hunt path is an ascii string that is used as a search path from a hunting process to a requested process. The first part specifies a unique name of a remote system. The second part contains the name of a process. Case is significant. The two parts are divided by a slash (/). The hunt path is used with the OS_hunt REOS system call, see ref. [2] for system call details.

Example: rp17/myprocess

The hunt path is used in analogy with an IP address and a UDP/TCP port number. The IP address specifies which remote system to communicate with, and the port number directs the communication to a specific process in that system.

Nr — No.		25/155 19-ANZ 212 08 Uen
Datum — Date	Rev	File
1999-11-16	A	

Link Handler An OSE Delta concept that allows separate systems to communicate with each other. A link handler is a process that manages logical connections in a multi-CPU system. A link handler responds to a hunt request and creates phantom processes when the hunted process reside in another CPU. A phantom process is a dummy representation of a remote process. This makes communication between processes in separate system transparent and done in the same manner as if they were in the same target.

Logical Channel A link between remote processes that is managed by a link handler.

Phantom Process An empty process with an optional redirection table. It does not contain any code and is therefore unable to receive any signals.

Redirection Table A table defining signal routes to receivers. Processes may be equipped with redirection tables.

RP Address A unique value assigned to an RP when defined in the CP. The RP address is used to name the Ethernet link handlers in an RP.

Nr — No.		25/155 19-ANZ 212 08 Uen
Datum — Date	Rev	File
1999-11-16	A	

3 GENERAL SERVICE INFORMATION

3.1 Service Description

The RP Interprocessor Communication function is implemented using the OSE Delta mechanism for building distributed systems. This gives an opportunity to create applications with functionality spread over several RPs. The OSE Delta mechanism is based on the link handler concept that handles logical channels between processes in separate targets.

For communication between processes in different systems, there has to be a link handler process in every system. This link handler manages logical channels between processes in different systems. A link handler responds to a hunt request and establishes a transparent logical channel to the specified system and process.

Communication through link handlers involves several steps as described below:

- Hunt** The **OS_hunt** REOS system call is used to search for a process and to obtain the process identity for that process. The link handler is responsible for establishing a logical channel to the desired process. A phantom process is created in the target of the hunter.
- Attach** The **OS_attach** REOS system call is used to make communication between processes safer. If process A attach to process B, process A will be notified if something happens to process B or if the logical channel between the communicating processes can not be maintained.
- Send or Receive** The ordinary REOS system calls **OS_send** and **OS_receive** are used to exchange signals between the processes.
- Detach** The **OS_detach** REOS system call is used to tell the operating system kernel that there is no longer any need to know if anything happens to the channel or the attached process.

Note that the attach mechanism is optional but should be used to ensure a safe and fault tolerant communication.

This service is completely accessed by using REOS or OSE Delta system calls as described below. See ref. [2] and [3] for more information.

Please note that the OS does not provide any service to find out what other RPs exist in the system. It is an application configuration issue to find out which system to communicate with.

3.2 Service Exception Handling

Applications using the “attach” method described below will be notified when there is a communication link failure.

Nr — No.		
25/155 19-ANZ 212 08 Uen		
Datum — Date	Rev	File
1999-11-16	A	

3.3 Service System Recovery

This interface is not affected by AXE system recovery.

If an RP restarts, processes in other RPs that have attached to processes in the restarted RP will be notified that the attached process is no longer available.

The interface is valid for an application at all times.

3.4 Service Characteristics

3.4.1 Link handler names

The link handlers in a system are given unique names that are used to find that system a multi-CPU environment. There are two types of names. A general name that does not include which Ethernet interface to be used and one that specifies a particular Ethernet interface.

The ordinary and simple way of using the link handler service is to use the general link handler name. Applications who do not want to be aware of the two physical communication links, uses this name. If there is a link failure, the application can be notified and search for its peer with the same hunt path again. Because of the Ethernet redundancy, a link will be setup over the other Ethernet instead.

There are also link handler names with an a/b suffix, which directs the traffic to a specific Ethernet interface. If a link goes down the hunt path must be changed in order to find the remote process again. Another usage of link specific link handler names could be to achieve load sharing between the two Ethernets. The application is responsible for directing individual signal transactions to one link or the other.

The following names are given to the link handlers in one RP:

"rp<RP Address>" example: rp145

"rp<RP Address>a" example: rp145a

"rp<RP Address>b" example: rp145b

3.4.2 Stand alone configuration

If the RP is used in a stand alone test environment without a CP, the RP address must be specified with the PBOOT board parameter **rp_address**. Remember that it must be unique in the communication domain.

Example using the PBOOT command interface:

```
>>set parameter rp_address 145
>>save parameters
```

Nr — No.		25/155 19-ANZ 212 08 Uen
Datum — Date	Rev	File
1999-11-16	A	

3.4.3 Restrictions

- The maximum signal size that can be sent remote is 65500 octets.

3.4.4 Characteristics

The link handler protocol stack is included in the operating system and provides the following services:

- Reliable data transfer. Lost signals are re-transmitted.
- Automatic fragmentation and re-assembly.
- Automatic detection of new systems.
- Automatic detection of link failure.

Automatic detection of new systems means that a few seconds (~2s.) after power on a new system, it appears in the name space for the other systems.

Automatic detection of link failure means that a few seconds after a link is considered dead, all users are notified if they have subscribed for that information.

- The communication area is limited by the broadcast domain on the Ethernet. The broadcast domain normally ends in a router or a host connected to the same Ethernet. That means that a host (workstation) could participate in the communication network.
- The communication service is point-to-point, unicast. Multicast or broadcast is not supported although broadcast is used internally in the link handler protocol stack.
- The link handler protocol stack uses the following type values for the Ethernet frames:

General Name 0x80c4, Banyan VINES IP

Specific Name 0x80c5, Banyan VINES Echo

Nr — No.		
25/155 19-ANZ 212 08 Uen		
Datum — Date	Rev	File
1999-11-16	A	

4 USE CASES

Sending and receiving signals are done exactly the same way as when communicating with a CPU-local process and are therefore not described in this document.

In the following examples, REOS system calls are used. Please refer to [2] for a detailed description of REOS system calls. OSE Delta system calls are described in [3].

4.1 Search for a process

The system call `OS_hunt` is used to search for a process. This system call can be used in a number of ways, below is a synchronous, “wait-forever” way used.

Example:

```
#define MY_HUNT_SIG 26316956
struct my_hunt_sig
{
    SIGSELECT sig_no;
};

static PROCESS find_process(char* name)
{
    union SIGNAL* sig;
    SIGSELECT hunt_sel[] = {1, MY_HUNT_SIG};
    PROCESS pid_;

    sig = OS_alloc(sizeof(struct my_hunt_sig), MY_HUNT_SIG);
    OS_hunt(name, 0, 0, &sig);
    sig = OS_receive(hunt_sel);
    pid_ = OS_sender(&sig);
    OS_free(&sig);

    return pid_;
}

static void somefunc(void)
{
    ...
    PROCESS pid_ = find_process("rp17/file_server");
    ...
}
```

4.2 Attach a signal to a process

The system call `OS_attach` is used in defensive programming to detect and resolve unexpected situations that occur when processes die or a communication link goes down. The attach system call can only be used if the process

Nr — No.		
25/155 19-ANZ 212 08 Uen		
Datum — Date	Rev	File
1999-11-16	A	

identity of a process is known. The process identity can be obtained with the hunt mechanism.

Example:

```
#define MY_ATTACH_SIG 26316957
struct my_attach_sig
{
    SIGSELECT sig_no;
};

static PROCESS find_and_attach(char* name, W32* attref)
{
    union SIGNAL* sig;
    PROCESS pid_;

    sig = OS_alloc(sizeof(struct my_attach_sig), MY_ATTACH_SIG);
    pid_ = find_process(name);
    *attref = OS_attach(&sig, pid_);

    return pid_;
}
```

4.3 Detach a signal from a process

The system call `OS_detach` is used to remove a signal previously attached by the caller. The detach system call can only be used after a attach system call.

Example:

```
static void foo(void)
{
    W32 attref;
    PROCESS pid_ = find_and_attach("rp19b/file_server", &attref);
    ...
    OS_detach(&attref)
    ...
};
```

4.4 Notification of logical channel failure

If something happens to a process or if the logical channel between the communicating processes can not be maintained, the previously attached signal will be returned in the same way as other signals. The notification means that it is no longer possible to send signals to the process the received signal was attached to.

5 SIGNAL IDENTIFICATIONS

If the `sig` parameter in the `OS_attach` call is set to zero, the kernel automatically allocates a signal with signal number 252. Please note that this signal

Nr — No.		25/155 19-ANZ 212 08 Uen
Datum — Date	Rev	File
1999-11-16	A	

number is in the predefined CP-RP range which could cause a conflict and therefore it should be avoided to used it.

6 TERMINOLOGY

CP	AXE Central Processor
IP	Internet Protocol
RP	AXE Regional Processor
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

7 REFERENCES

- [1] OSE Real Time Kernel, Users Guide; Enea OSE Systems AB; Document No: 420e/OSE93-2
- [2] REOS OPERATING SYSTEM INTERFACE; 6/155 19-ANZ 212 08
- [3] OSE Real Time Kernel, Reference Manual; Enea OSE Systems AB; 420e/OSE93-1