Building the Ingex tapeless recording software

12 February 2016

Table of Contents

Installing openSUSE

General

These instructions apply to the 64-bit version of openSUSE 13.2. Deselect the autoconfiguration option to get more control over the installation.

Partitions

It's best to have a separate RAID for storing video files. We generally have first disc as system disc (no separate partition for /home) and then 3 discs in RAID0, using XFS, mounted on /video for recordings.

Desktop

Choose the KDE desktop.

Software packages

It's also more convenient to include the following packages during installation by clicking on the heading “Software” when you reach the screen “Installation Settings”:

Server Functions

- File server
- Web and LAMP server

Development

- C/C++ development
- Perl development
- Linux Kernel Development

Firewall and ssh

You may wish to enable ssh and disable the firewall if using multiple machines in a network.

Online update

At the end of the installation, you should accept the Online update option.

Before you start

Set up new user

It will help to go into YaST->Security and Users -> User Management and create a new user named 'ingex' as this gives a clean area in which to work.

It's also be convenient to allow the user *ingex* to be able to use the “sudo” command to run commands with root privileges. You can enable this in YaST -> Security and Users ->sudo by adding a rule as follows:

- User: ingex
- Host: ALL
- RunAs: leave blank
- No Password: ticked
- Commands to run: ALL

Terminology

The styling used in this document is that each example command line starts with a prompt string such as **ingex**> which mean the command such be entered as the user ingex. The alternative is **root**> which means you should be root to enter this command. Remember that this prompt is not part of the command; the command starts after the '>'.

Graphics Card

It is recommended to install any graphics card early on. Nvidia drivers may update the kernel and so it's important to make sure kernel-sources are updated as well, and that this is all done **before** the DVS installation processes.

See instructions at: http://en.opensuse.org/SDB:NVIDIA_drivers

RAID

We recommend a small number (e.g. 3) of drives configured as RAID0 for video recording and replay. Use a filesystem such as XFS and mount on /video

Downloading the Ingex software

There are ways to download the Ingex source code from SourceForge:

- CVS (updated regularly as development proceeds)
- File release (more stable versions)

CVS

Login as user ingex, open a terminal window and start by making a local directory for the software.

```
ingex> mkdir ~/ap-workspace
ingex> cd ~/ap-workspace
```

Now download the Ingex software from the CVS respository on Sourceforge, with these two commands:

First login with this command. When prompted for a password, just press enter.

```
ingex> cvs -d:pserver:anonymous@ingex.cvs.sourceforge.net:/cvsroot/ingex login
```

Next get the ingex source code with the checkout command:

```
ingex> cvs -z3 -d:pserver:anonymous@ingex.cvs.sourceforge.net:/cvsroot/ingex co -P ingex
```

This will download the entire project.

Note that if you are behind a firewall, you will probably need to install dante, edit /etc/socks.conf and precede the cvs commands with socksify. (See troubleshooting below)

Throughout these instructions, the term "$workspace" is used to refer to the directory where you have the ingex-studio source files (studio, player directories etc.), either from CVS or from a file release. Either just bear that in mind when typing the commands or set this as an environment variable if you wish.

For example, with CVS:

```
ingex> export workspace=/home/ingex/ap-workspace/ingex
```

Or for example, with a file release:

```
ingex> export workspace=/home/ingex/ingex-studio-1.0.0
```

It would be sensible to put this line into your .bashrc file (i.e. ~/.bashrc), so it's always set.

Troubleshooting

If the CVS download fails because you are behind a proxy server, add the proxy details to the cvs command, as in the next example, but using the IP address and port number for your proxy server:

```
ingex> cvs "-
d:pserver;proxy=192.168.1.255;proxyport=80:anonymous@ingex.cvs.sourceforge.net:/cvsroot/in
gex" login
```

```
ingex> cvs -z3 "-
d:pserver;proxy=192.168.1.255;proxyport=80:anonymous@ingex.cvs.sourceforge.net:/cvsroot/in
gex" co -Pd ingex
```

As an alternative to using the proxy option in the cvs command, you can instead use socksify (provided when you install dante).

Updating

To incorporate updates into your existing installation, run the following command:

```
ingex>cvs -z3 update -Pd
```

File release

Go to http://sourceforge.net/projects/ingex/files/ and download all the files for the release version in question.

Uncompress and extract the ".tgz" files

Installation Prerequisites

There are five main sets of packages to install before building the Ingex software:

- additional standard packages and perl modules
- pre-built RPMs of other packages used by Ingex
- DVS SDK
- AAF SDK
- Javascript Ext library
- Flowplayer Flash

Not all of these are required for each part of the Ingex software, so if you wish to build the modules separately and want a minimum installation, refer to their separate build instructions.

Standard Packages

Open YaST and under Software Management, install (together with any dependencies):

```
libjpeg62-devel
libbz2-devel
portaudio-devel
postgresql-server
postgresql-devel
libpqxx-devel
libXerces-c-devel
wxWidgets-devel
libSDL-devel
liburiparser-devel
```

If you selected the software options suggested in "Installing SUSE" during installation, then the following packages should already be installed. Install them now if YaST indicates they're not present.

```
autoconf
automake
kernel sources
libopenssl-devel
libtool
libuuid-devel
libuuid1
fontconfig-devel
freetype2-devel
```

You may also find the following packages useful:

```
phpPgAdmin
gkrellm
dante
```

acroread

Perl modules

The following perl modules should be installed:

Module	**Source**
CGI::Session	YaST
Clone	YaST
common::sense	YaST
DBD::Pg	YaST
DBI	YaST
JSON::XS	YaST
Linux::Inotify2	YaST
Log::Dispatch	YaST
Log::Log4perl	YaST
Switch	YaST
Template::Toolkit	YaST
Text::Template	YaST
XML::Simple	YaST
Filesys::DfPortable	CPAN
IPC::ShareLite	CPAN
Log::Handler	CPAN
PDF::Create	CPAN
Proc::Daemon	CPAN
Term::ANSIColor	CPAN

There are several ways of installing these.

i) YaST

This is recommended where the modules are available. The package naming convention translates the :: into a dash, and prepends perl-, so that, for instance, XML::Simple appears as perl-XML-Simple.

ii) Using the cpan script

```
ingex> su
# cpan <module name> <module name> ...
```

iii) Manually

See the instructions at www.cpan.org/modules/INSTALL.html but, to summarise:

Get each file, then

```
ingex> tar xvf <module>.tar.gz
ingex> cd <module>
ingex> perl Makefile.PL
ingex> make
ingex> make test
ingex> sudo make install
```

Pre-built RPMs

i) ACE and TAO

Communications between the Ingex recorder software and its controller is implemented using ACE (the "Adaptive Communications Environment") and TAO ("The ACE ORB"). Full details of these packages are available from www.cs.wustl.edu/~schmidt/ACE.html and extensive documentation can be found there.

We currently recommend the RPMs provided on the Ingex SourceForge website.

First make a directory for them:

```
ingex> mkdir ~/rpms
ingex> cd ~/rpms
```

The files are available from SuSE at
http://download.opensuse.org/repositories/devel:/libraries:/ACE:/micro/

Choose the appropriate folder for you machine operating system and architecure.

– i586/	32-bit
– x86_64/	64-bit

Download the following RPMs into the created directory.

These are examples for a 64-bit build (NB: the files may have been updated since these notes were written):

● ace	ace-6.0.2-51.x86_64.rpm
● ace-devel	ace-devel-6.0.2-51.x86_64.rpm
● ace-gperf	ace-gperf-6.0.2-51.x86_64.rpm
● ace-kokyu	ace-kokyu-6.0.2-51.x86_64.rpm
● ace-xml	ace-xml-6.0.2-51.x86_64.rpm
● mpc	mpc-6.0.2-51.x86_64.rpm
● tao	tao-2.0.2-51.x86_64.rpm
● tao-cosnaming	tao-cosnaming-2.0.2-51.rpm
● tao-devel	tao-devel-2.0.2-51.rpm

Then install them all **at once** with this command:

```
ingex> sudo rpm -i *
```

Note that you will probably get a message along the lines of:

```
  warning: ace-6.0.2-51.x86_64.rpm: Header V3 DSA/SHA1 Signature, key ID 0131d685:
NOKEY
```

To finish setting the ACE/TAO environment variables you **must** log-out and then back in to continue with the installation process.

ii) Additional RPMs

Download and install the following RPMs from the Ingex Sourceforge website:

- codecs-for-ffmpeg
- ffmpeg-DNxHD
- shttpd

Install them with these commands, making sure that you install the codecs-for-ffmpeg rpm first, otherwise the ffmpeg rpm will complain about missing dependencies. (XXXXX represents each file's version number; choose the latest found on the website.)

```
ingex> sudo rpm -i codecs-for-ffmpeg-XXXXX.rpm
ingex> sudo rpm -i ffmpeg-DNxHD-XXXXX.rpm
ingex> sudo rpm -i shttpd-XXXXX.rpm
```

DVS software installation

If you have DVS cards installed, you should now install the DVS software for them by following the instructions in the later section "DVS card drivers and SDK".

Blackmagic software installation

The software includes limited experimental support for Blackmagic DeckLink cards. If you want to use this, see the instructions in the later section "Blackmagic card drivers and SDK".

AAF software installation

First make a new directory for the source code of the AAF toolkit and move into it. It's best to keep this separate from the Ingex directories, as it's an independent, self-contained, suite of software.

```
ingex> mkdir ~/AAFtoolkit
ingex> cd ~/AAFtoolkit
```

Checkout the AAF sourcecode from Sourceforge:

```
ingex> cvs -d:pserver:anonymous@aaf.cvs.sourceforge.net:/cvsroot/aaf co -P AAF
```

and make the package

```
ingex> cd AAF
ingex> make install DISABLE_FFMPEG=1
```

It's convenient to set an environment variable AAFSDKINSTALL so that we can easily pick-up the AAF installation location later. Use the command here, substituting the installation directory (i.e. the directory where you ran the tar command above), where it says <dir>.

FOR 64bit:

```
ingex> export AAFSDKINSTALL=<dir>/AAF/AAFx86_64LinuxSDK/g++
```

FOR 32bit:

```
ingex> export AAFSDKINSTALL=<dir>/AAF/AAFi686LinuxSDK/g++
```

As with the earlier environment variable, it's helpful to add this line to your ~/.bashrc file, so the value is always set.

BMX library

NB. Not needed when using file release 1.0.0

See http://sourceforge.net/p/bmxlib/home/

```
ingex> mkdir bmx
ingex> cd bmx
ingex> git clone http://git.code.sf.net/p/bmxlib/libmxf libMXF
ingex> git clone http://git.code.sf.net/p/bmxlib/libmxfpp libMXF++
ingex> git clone http://git.code.sf.net/p/bmxlib/bmx bmx
ingex> cd libMXF
ingex> ./autogen.sh && ./configure && make && make check && sudo make install
ingex> cd ../libMXF++
ingex> ./autogen.sh && ./configure && make && make check && sudo make install
ingex> cd ../bmx
ingex> ./autogen.sh && ./configure && make && make check && sudo make install
```

NB. An alternative command that may work better with older versions of git is:

```
ingex> git clone git://git.code.sf.net/p/bmxlib/libmxf libMXF
```

To update the installation:

```
ingex> cd libMXF
ingex> git pull
ingex> make && make check && sudo make install
```

And the same in libMXF++ and bmx directories.

Web root

See section "Configuring the web server" for explanation of 'web root'. Typically this is /srv/www for SuSE and /var/www for Debian distributions. Set an environment variable to hold this:

```
ingex> export webroot=/srv/www
```

Javascript Ext library

Download and install the Ext Javascript library, using the following steps.

Download a copy of Ext JS 4.1.1a, e.g. from http://olex.openlogic.com/packages/extjs/4.1.1a
N.B. The more recent 5.1.0 version does not seem to work with the Ingex code.

Unzip the file.
Make directory $webroot/htdocs/ingex if it doesn't already exist.
Ensure $webroot/htdocs/ingex/ext does not exist.

```
ingex> sudo mv ext-4.1.1a $webroot/htdocs/ingex/ext
```

Flowplayer Flash

Next download and install Flowplayer Flash and associated plugins.

```
ingex> mkdir /tmp/flowplayer
ingex> cd /tmp/flowplayer/
ingex> wget http://releases.flowplayer.org/flowplayer/flowplayer-3.2.15.zip
ingex> unzip flowplayer-3.2.15.zip
ingex> cd flowplayer
ingex> wget http://releases.flowplayer.org/flowplayer.content/flowplayer.content-3.2.8.swf
ingex> wget
http://releases.flowplayer.org/flowplayer.pseudostreaming/flowplayer.pseudostreaming-
3.2.11.swf
ingex> wget http://releases.flowplayer.org/flowplayer.audio/flowplayer.audio-3.2.10.swf
ingex> sudo mkdir -p $webroot/htdocs/ingex/review/lib/flowplayer
ingex> sudo mv flowplayer*.min.js
$webroot/htdocs/ingex/review/lib/flowplayer/flowplayer.min.js
ingex> sudo mv flowplayer-*.swf $webroot/htdocs/ingex/review/lib/flowplayer/flowplayer.swf
ingex> sudo mv flowplayer.audio*.swf
$webroot/htdocs/ingex/review/lib/flowplayer/flowplayer.audio.swf
ingex> sudo mv flowplayer.content*.swf
$webroot/htdocs/ingex/review/lib/flowplayer/flowplayer.content.swf
ingex> sudo mv flowplayer.controls*.swf
$webroot/htdocs/ingex/review/lib/flowplayer/flowplayer.controls.swf
ingex> sudo mv flowplayer.pseudostreaming*.swf
$webroot/htdocs/ingex/review/lib/flowplayer/flowplayer.pseudostreaming.swf
ingex> cd
ingex> rm -r /tmp/flowplayer
```

Final check

You should have added two new 'export' lines in your ~/.bashrc file, i.e.

```
export DVSSDK=<path to your DVS SDK>
```

(if using DVS cards), and

```
export AAFSDKINSTALL=<path to your AAF SDK>
```

and with $ACE_ROOT too, those environment variables should now be set. It's sensible to check them before proceeding with the build, which you can do as follows:

First log out, and log back in again to pick up the ACE_ROOT setting, and then type:

```
ingex> echo $ACE_ROOT
```

and you should see:

```
/usr/share/ace
```

Then check the other two with:

```
ingex> echo $DVSSDK
ingex> echo $AAFSDKINSTALL
```

and these should indicate the correct paths to each package, as set earlier in this section.

Next stages

With the pre-requisites installed, you can now build install the Ingex software from the source code, as described in the section "Building the full system".

DVS card drivers and SDK

The current implementation of Ingex works with DVS capture cards, and so you will need the software for the cards from DVS.

If you do not have the DVS software at this stage then you can test a recorder instead by using the "dvs_dummy" program that will be created in ingex/studio/capture. In that case, skip the instructions here covering the installation of the DVS software and its configuration.

DVS SDKs are available to registered users here (follow link for your board):

http://www.dvs.de/products/video-boards.html

Make a directory for the dvs software:

```
ingex> mkdir ~/dvs
ingex> cd ~/dvs
```

Note: It is recommended that the latest firmware is used for any DVS hardware.

Download the linux SDK, e.g. sdk4.3.5.10.tar.gz, into this directory.

Extract the SDK files from the tar file:

```
ingex> tar xf sdk4.3.5.10.tar.gz
```

As other parts of the build process will need to access the DVS SDK later, we need to set an environment variable with the path details.

```
ingex> export DVSSDK=<location of your DVS SDK>
```

e.g. `export DVSSDK=/home/ingex/dvs/sdk4.3.5.10`

It is also sensible to put this line into your .bashrc file (i.e. ~/.bashrc), so it's always set.

Before building the DVS driver, you need to configure the kernel.

Do these next commands as root, so su to root:

```
ingex> sudo su
```

Change to the directory for your kernel sources.

```
root> cd /usr/src/linux
root> make oldconfig
root> make scripts/mod/                #(Note - the trailing slash is important)
```

Exit the root shell:

```
root> exit
```

Now we can make the DVS driver:

FOR 32bit:

```
ingex> cd $DVSSDK/linux-x86/driver
ingex> ./driver_create
```

FOR 64bit:

```
ingex> cd $DVSSDK/linux-x86_64/driver
ingex> ./driver_create
```
< various build messages will appear >

Note that for certain SDK and OS versions you may need to modify a couple of files for the build to work.

If necessary, add the following near the beginning of `src_v4r3/driver/linux.c` and `src_lucy/driver/linux.c`

```
#ifndef init_MUTEX
#define init_MUTEX(_m) sema_init(_m,1)
#endif
```

```
ingex> sudo ./driver_load
```

If this is successful, you will see messages like these appearing:

```
- load dvsdriver
- load dvsdriver_sdio
Created device: /dev/iris0 252 0
Created device: /dev/iris1 252 1
```

The DVS driver is now ready.

Driver loading at boot time

Although the driver can be loaded manually, as above, with the driver_load command, it's convenient for this to be done automatically at boot time. The file dvs_setup deals with this.

Move to the ingex scripts directory:

```
ingex> cd $workspace/studio/scripts
```

Edit the file dvs_setup, using your preferred editor, in this example, vim:

```
ingex>  vim dvs_setup
```

and change the following line to be the path of the DVS SDK, just installed, e.g:

```
DVS_DIR=/home/ingex/dvs/sdk4.3.5.10
```

Save the change and close the editor.

You can test the file by using this command

```
ingex> sudo ./dvs_setup start
```

There will be a pause, whilst this loads the driver for the cards.

To make this automatic at boot time, follow these steps:

Copy the dvs_setup file to /etc/init.d

```
ingex> sudo cp dvs_setup /etc/init.d
```

and make it start at boot time:

```
ingex> sudo /sbin/chkconfig -a dvs_setup
```

Note for those running a prior installation of Ingex: the DVS card configuration which was previously also done by dvs_setup, is now carried out when the capture program dvs_sdi is run.

Checks

1. You did export DVSSDK didn't you? Check that

```
ingex> echo $DVSSDK
```

displays the correct location of the SDK.

2. Check the DVS driver is working with the command

FOR 32bit:

```
ingex> cd $DVSSDK/linux-x86/bin
ingex> ./svram info
```

FOR 64bit:

```
ingex> cd $DVSSDK/linux-x86_64/bin
ingex> ./svram info
```

3. Check the DVS licence keys are installed

```
ingex> SCSIVIDEO_CMD=PCI,card=0 ./svram licence show
ingex> SCSIVIDEO_CMD=PCI,card=1 ./svram licence show
```

To install a licence key (which is stored on the DVS card, not the computer's hard disk):

```
ingex> SCSIVIDEO_CMD=PCI,card=0 ./svram licence key1 000000000000000000000000
```

Blackmagic card drivers and SDK

The current software includes limited support for Blackmagic Decklink cards by means of an alternative capture program "bmd_sdi". To build this, you will need (a) the SDK and (b) the drivers and library.

Note that the Ingex Player, and therefore the controller Ingexgui, does not support playback to Blackmagic cards at present.

SDK

This is basically a set of include files. You can download it from:
https://www.blackmagicdesign.com/support/family/capture-and-playback

Unzip it to a suitable location and set the enviroment variable BMD_HARWARE_INCLUDE to the location of the linux/include directory.

For example:

```
ingex> export BMD_HARDWARE_INCLUDE=/home/ingex/BlackmagicDeckLinkSDK/Linux/include
```

Drivers and library

Download "Desktop Video for Linux" from:
https://www.blackmagicdesign.com/support/family/capture-and-playback

Install DKMS, e.g. using http://linux.dell.com/dkms/permalink/dkms-2.2.0.3-1.noarch.rpm

Extract the Desktop Video Linux tar file:

```
ingex> tar xf Blackmagic_Desktop_Video_Linux_10.5.4.tar
```

In the appropriate rpm directory, e.g. rpm/x86_64, install the non-gui RPM ignoring dependencies:

```
root# rpm -ivh --nodeps desktopvideo-10.5.4-a4.x86_64.rpm
```

Install driver:

```
root# modprobe blackmagic
```

Check/update card firmware:

```
root# BlackmagicFirmwareUpdater status
root# BlackmagicFirmwareUpdater update
```

Reboot.

Building the full system

You can build all the software by following the instructions in this section. The procedure is slightly different, depending on whether the source code came from CVS or from a file release. If you need to build any modules separately, then instead follow the steps in the later section covering that module.

CVS

Build and install YUVlib

```
ingex> cd $workspace/YUVlib
ingex> make
ingex> sudo make install
```

NB. libMXF and libMXF++ are now provided by BMX library

Move to the top level directory containing the Ingex software, i.e.

```
ingex> cd $workspace
```

and run the 'make'...

```
ingex> make
```

Build messages will appear during this process, there will probably be some warnings but hopefully no errors.

When the above 'make' has finished, you need to install one of the executables, as follows:

```
ingex> cd $workspace/player/ingex_player
ingex> sudo make install
```

(note: if an error about “no DVS available” occurs here ignore it)

You can optionally install another, as follows:

```
ingex> cd $workspace/studio/ace-tao/Ingexgui
ingex> sudo make install
```

(This will allow you to run the controller, ingexgui, without specifying its path.)

File release

Move to the directory containing YUVlib

```
ingex> cd YUVlib-src-1.0.0
```

and run the 'make' and install

```
ingex> make
ingex> sudo make install
```

Move to the directory containing libMXF and libMXF++

```
ingex> cd libMXF-src-1.0.0
```

and run the 'make' and install

```
ingex> make
ingex> sudo make install
```

Move to the directory containing Ingex-Studio

```
ingex> cd ingex-studio-src-1.0.0
```

and run the 'make'...

```
ingex> make
```

Next steps

The Ingex software is now built but needs to be configured before you can run it. To do this, you should now go through these sections:

- Configuring the Naming Service
- Configuring the database
- Configure the web server
- Configure the Controller

Once these are done, you can follow the section "Running the System", to get everything working.

To use some of the additional features of Ingex, you should subsequently look through the sections:

- Transfer Server (for background copying from recorder to a remote destination)
- Director's Cut (for logging cuts made during recording and creating edits from them)
- Multicasting the video (for distributing the live video feeds from Ingex)

OS Configuration

Under openSUSE 12.x you need to adjust the maximum shared memory segment size.

Edit `/etc/sysctl.conf` and add the following:

```
kernel.shmmax = 1073741824
```

Configuring the CORBA Naming Service

The Naming Service needs to run on one PC in the system so that as each Ingex recorder starts, it can register itself with the Naming Service, and hence becomes available to any controller. You need to configure the Naming Service and enable it as a service to start at boot time as described here.

Remember that you only need to do this on one PC in an Ingex setup.

Configure the Naming Service

On the PC that's going to run the Naming Service, using your preferred editor, edit the file /etc/tao/tao-cosnaming, e.g.

```
ingex> sudo vim /etc/tao/tao-cosnaming
```

Edit the line beginning TAO_COSNAMING_ENDPOINT to:

```
TAO_COSNAMING_ENDPOINT="-ORBEndPoint iiop://<your IP address>:8888
-ORBDottedDecimalAddresses 1"
```

where <your IP address> is the IP address of the PC which will run the Naming Service. For example:

```
TAO_COSNAMING_ENDPOINT="-ORBEndPoint iiop://192.168.1.181:8888
-ORBDottedDecimalAddresses 1"
```

Rather than using an IP address, you can instead use the hostname at that point, but in that case omit the -ORBDottedDecimalAddresses 1 arguments. Also, to avoid problems later, make sure the DNS entry is correct by pinging the hostname from elsewhere.

If you are using fixed IP addresses, then it's preferable to use the IP address.

Also comment out the line beginning TAO_COSNAMING_PERSISTFILE as persistent naming service data is only likely to cause confusion.

Save the file and close it.

Enabling the Naming Service

```
root# systemctl enable tao-cosnaming
root# systemctl start tao-cosnaming
```

Configuring the Database

The database is required on only one PC in an Ingex setup. This can be a PC running the Ingex recorder itself, or a separate PC. It is used to store the configurations of the Ingex recorders and also details of the recordings that are made.

Prerequisites

If you have followed the earlier section “Installation Prerequisites”, you can skip this step. Otherwise, in YaST, make sure the following is installed:

  postgresql-server

Setting up the database

Start the postgres service. As you need to do this as root, first su

```
ingex> sudo su
```

then start the database as root

```
root> /etc/init.d/postgresql start
```

If this fails to start, then you might want to switch on logging as described in the document www.postgresql.org/docs/8.1/static/logfile-maintenance.html

Starting postgresql will create an initial database for you in "/var/lib/pgsql/data" - move to that directory (still as root)

```
root> cd /var/lib/pgsql/data
```

Next edit the file "pg_hba.conf" (the PostgreSQL Client Authentication Configuration File) and add entries to allow access for the hosts which need to access the database. In this case, we'll allow access from all hosts. First open the file in an editor (in this example, vim)

```
root> vim pg_hba.conf
```

Towards the end of the file, below the heading "Put your actual configuration here", you will see these entries

```
# TYPE  DATABASE    USER        CIDR-ADDRESS          METHOD
# "local" is for Unix domain socket connections only
local   all         all                               peer
# IPv4 local connections:
host    all         all         127.0.0.1/32          ident
# IPv6 local connections:
host    all         all         ::1/128               ident
```

Change the "METHOD" entry for each line from "ident" to "trust", so the entries now look like this:

```
# "local" is for Unix domain socket connections only
local   all         all                               trust
# IPv4 local connections:
host    all         all         127.0.0.1/32          trust
# IPv6 local connections:
host    all         all         ::1/128               trust
```

To allow access to the database from the other PCs in your Ingex system, add the following line but, if necessary, with an appropriate IP address range depending on your network configuration, instead of 172.29.144.0/20 :

```
host     all          all    172.29.144.0/20        trust
```

Save the changes and close file.

Next edit the file "postgresql.conf"

```
root> vim postgresql.conf
```

Find the line #listen_addresses = 'localhost' and change this to allow access from the required hosts. Delete the # and change localhost to * to make the line read:

```
listen_addresses = '*'
```

Save the changes and close file.

Re-start the postgres service using the command:

```
root> /etc/init.d/postgresql restart
```

For convenience, it's best to make the postgres service start at boot time. To do this in YaST:

- start YaST and select *System*, then *System Services (runlevel).*
- Find the entry *postgresql* and enable it.
- Click on *Expert* and ensure that it is set to start at run levels 3 and 5.

Alternatively, you can use this command:

```
root> /sbin/chkconfig -a postgresql
```

Exit the root shell, as the further command are done as the Ingex user

```
root> exit
```

Create the database

Now we create the database that will hold the details of the recorder configurations and also the recordings that are made. In this example, we'll call it "prodautodb".

Remember that, as shorthand, $workspace represents the directory holding the Ingex studio (and other) directories.

Move to the directory containing the setup scripts:

```
ingex> cd $workspace/studio/database/scripts
```

Next run the three scripts that will setup the database user and an initial database and then set a basic configuration for the recorders:

```
ingex> ./create_bamzooki_user.sh
ingex> ./create_prodautodb.sh
ingex> ./add_basic_config.sh
```

If anything goes wrong at this point, or you simply want to start again later, you can delete the database with the command:

```
ingex> ./drop_prodautodb.sh
```

and then repeat the *create_prodautodb.sh* and *add_basic_config.sh* commands above to start with a clean installation of the database.

Troubleshooting

If you get the error message Ident authentication failed for user "postgres" then check the file /var/lib/pgsql/data/pg_hba.conf which you edited above when setting up the access permissions to the database. The "ident sameuser" entries must be changed to "trust".

The database is now ready. You can check it manually with the psql command:

```
ingex> psql prodautodb bamzooki
```

At the prompt **prodautodb=>** that appears, perform a simple access to the database with this command (don't forget the ; at the end of the command):

```
prodautodb=> select * from fileformat;
```

Some (slightly cryptic) file format details should appear showing that the database is working and has relevant entries.

Exit psql with \q or <control-d>:

```
prodautodb=> \q
```

Configuring the Web server

The web server provides a convenient method of accessing the configuration details for the recorders, monitoring their operation and displaying details of recordings that have been made.

It is probably most convenient to run the web server on the same PC as the Ingex database.

Web root

The first decision to make is where to install the web site. This depends on the 'web root' directory of your system. Typically this is /srv/www for SuSE and /var/www for Debian distributions. Set an environment variable to hold this:

```
ingex> export webroot=/srv/www
```

You should also change WEB_ROOT in your ingex.conf and review.conf files if not using the default setting of /srv/www

Prerequisites

If you have already installed the pre-requisite packages as part of a full system build, then you can skip this step.

In YaST, check that the following packages have already been installed:
  apache2

Perl also requires all the modules listed in the 'Installation Prerequisites' section (such as XML::Simple, PDF::Create and others). See the instructions there for information on installing the modules. If you have already installed them as part of a full system build, you will not need to do it again.

Installation

Move to the directory containing the web software.

```
ingex> cd $workspace/studio/web/WebIngex
```

and install the files with the command:

```
ingex> sudo ./install.sh $webroot
```

Note: if you are updating the installation, you can run update.sh instead of install.sh and this will preserve your previous configurations.

Now edit the configuration file for the web software:

```
ingex> sudo vi $webroot/cgi-bin/ingex-config/WebIngex.conf
```

Edit the file to match your system. Make sure your database name and access details are correct in the "database connection settings" section shown below. Note that if your database is not on the same PC as you are now installing the web server, you will need to change the details in the line "db_host" by replacing 'localhost' with the name or IP address of the PC running the Ingex database.

```
# database connection settings
#
db_host = localhost
db_name = prodautodb
db_user = bamzooki
db_password = <password>
```

Save the file and exit the editor.

There are two methods of browsing and exporting the material: via a simple, single panel interface or a more flexible drag/drop mode. In order to switch between the two, edit the material page configuration file:

```
ingex> sudo vi $webroot/cgi-bin/ingex-modules/Material.ingexmodule/material.conf
```

and change the "drag_drop" parameter in "material.conf" to 'true' or 'false'.

```
#
# Tree material page settings
#
drag_drop = true
```

Note that the “Configure” section of the web interface provides another way of edting this file.

A separate configuration file is used by the web material review service. Edit this to match your system setup, paying particular attention to the system paths. The default settings will work on most systems:

```
ingex> sudo vi $webroot/cgi-bin/ingex-review/review.conf
#
# Ingex database connection settings
#

db_host = localhost
db_name = prodautodb
db_user = bamzooki
db_password = <password>

#
# System paths settings
#

html_root = /srv/www/htdocs/
cgi_root = /srv/www/cgi-bin/
url_root = /ingex/review/
url_browse = /browse/
browse_path = /srv/www/htdocs/browse/
thumbs_path = /srv/www/htdocs/browse/thumbs/
stills_path = /srv/www/htdocs/browse/stills/
mxf_path = /store/mxf_online/

#
# use secure logins? EXPERIMENTAL
#

http_auth = no

#
```

```
# Encode settings
#

encode_fmt = mp4
default_search_vfmt = 222
```

This configuration can later be changed graphically using the 'settings' part of the review web page.

Now make the create_aaf (if not already built) and install it by following these steps:

```
ingex> cd $workspace/studio/processing/createaaf
```

If you haven't built the full system already, build the createaaf executable now:

```
ingex> make
```

Next create a directory which can be reached by the web server, and copy the executable into it (example is for SuSE):

```
ingex> sudo mkdir -p /srv/ingex/createaaf/
ingex> sudo cp create_aaf /srv/ingex/createaaf/
```

As an alternative to the above two commands, on SuSE you can use:

```
ingex> sudo make install-web
```

You'll also need to copy a library file there:

```
ingex> sudo cp $AAFSDKINSTALL/bin/debug/libcom-api.so /srv/ingex/createaaf/
```

.. and a directory containing two extension library files too:

```
ingex> sudo cp -r $AAFSDKINSTALL/bin/debug/aafext /srv/ingex/createaaf/
```

The deletepkg executable now needs to be installed using the following steps:

```
ingex> cd $workspace/studio/processing/deletepkg/
```

Again, if the full system has not been built, build the deletepkg executable:

```
ingex> make
```

Next, create a directory which can be reached by the web server, and copy the executable into it:

```
ingex> sudo mkdir /srv/ingex/deletepkg/
ingex> sudo cp deletepkg /srv/ingex/deletepkg/
```

As an alternative to the above two commands, on SuSE you can use:

```
ingex> sudo make install-web
```

The web server makes use of a system log file where errors are written. Create this using the following command:

```
ingex> sudo touch /var/log/ingex.log
```

Now set permissions so the web server can write to it:

```
ingex> sudo chown wwwrun:www /var/log/ingex.log
```

Generating browse material (optional)

In order to preview recorded video material on the web server, a browse quality copy of the material must be encoded and stored on the web server.

The first step is to configure the Ingex recorder to output web format mov files to a browse material directory such as '/video/browse'. If running webserver on a separate fileserver, the recorder should also be configured to copy these to a suitable directory on the server, such as '/store/browse'. The process for this is explained in the section "Configuring Recorders". A suitable format is 'SD H264 2 Mbit/s and AAC'.

Depending on whether you are running webserver on recorder or fileserver (the latter is recommended), set an environment variable for browse material directory:

```
ingex> export browse_path=/video/browse
```
or
```
ingex> export browse_path=/store/browse
```

Make sure the directory exists:
```
ingex> mkdir -p $browse_path
```

Next create a web path which is symbolically linked to the browse directory and create directories for the output of thumbnails:
```
ingex> sudo ln -s $browse_path $webroot/htdocs/
ingex> sudo mkdir $webroot/htdocs/browse/thumbs
ingex> sudo mkdir $webroot/htdocs/browse/stills
ingex> sudo chown wwwrun:www $webroot/htdocs/browse/thumbs
ingex> sudo chown wwwrun:www $webroot/htdocs/browse/stills
```

Finally, edit the system apache configuration using the command:
```
ingex> sudo vim /etc/apache2/default-server.conf
```

Add the line 'Options Indexes FollowSymLinks' under <Directory "/srv/www/htdocs"> section, as below:

```
#
# Configure the DocumentRoot
#
<Directory "/srv/www/htdocs">
        Options Indexes FollowSymLinks

        AllowOverride None

        Order allow,deny
        Allow from all
</Directory>
```

Starting the Apache server

In YaST, go to System -> System Services (RunLevel). Select Apache2 in the list and click on Enable button. You can also check using "Expert Mode" that this will start automatically in run levels 3 and 5.

Click "Finish" to leave the System Services page.

Testing

Point your web browser at the page http://<your host name>/ingex and you should see the Ingex overview page.

If this doesn't work then open a browser on the same PC as the Apache server, and try http://localhost/ingex to check that the server is running.

Next, browse to the page http://<your host name>/ingex/review which should show the Ingex web review page.

If there are still problems, try http://<your host name>/ingex/test and follow the instructions to test your Perl, database and web-server setup.

Configuration Settings

Monitoring

With the web server running, you can now configure the monitoring facility within Ingex. However this is not required for initial testing and so this section can be left until later, if preferred.

Open the web interface and click 'configure' in the top right corner to configure the nodes on your network that are to be monitored.

Now specify which machines are to be monitored on your network. For each machine:

- Under *Node Configuration*, click 'Create New'. Enter the name and IP address (or host name) of the node to add. If the machine is the one running the web server, you can enter 'localhost', although using IP addresses may be more reliable.
- Select the type of machine (Recorder or Filestore) in order to specify which types of information to monitor (e.g. a file server has no capture cards to monitor)
- If you wish to monitor the free space of only some volumes (drives), specify them in a comma separated list, e.g. to monitor the space in /video and /data, you would enter: /video,/data - leaving this box blank will monitor all volumes.

At this stage, as nothing is running, if you go to the 'Status' page, you will be told that monitoring information cannot be displayed. This is because each machine to be monitored must have a program (or multiple programs) running which provide the monitoring data for the web server. As described later, on each recorder machine, you will need to run both *nexus_web* and *system_info_web*, and on a file server, just *system_info_web.* For information on running these, see "Start Monitoring" in the section “Starting additional facilities”.

Configuring Recorders

You will need to configure the recorder using the web page for Ingex configuration. As installed, this has a basic configuration but you will probably want to make some changes, e.g. recording formats, before use.

Note: See separate document “Ingex Modules and Configurations” for the detail of how to configure Ingex as there are a large number of options available. The description here is intended to be sufficient to get the system running and perform many of the basic operations.

If not yet present, first copy the ingex.conf file to either your home directory or to /etc – the one in your home directory will be used if it is present, otherwise the one in /etc; failing that, some built-in defaults will be used.

```
ingex> cp $workspace/studio/scripts/ingex.conf ~
```

You can edit this file to change startup options when using the startIngex.sh script.

If you have installed from a file release, you will want to change the line:
INGEX_DIR="/home/ingex/ap-workspace/ingex"

to something like:
INGEX_DIR="/home/ingex/ingex-studio-1.0.0"

Point your browser at:
http://<hostname of your web server>/ingex

and click on the Setup tab.

Go to the Recorder page using the link on the left of the screen.

The list of recorders will appear, with three entries; the first two of these have a choice of two configurations, while the third has one.

Recorder “Ingex” is set to use configuration “Ingex-MXF”. Click on this configuration name and scroll down the page that appears to see the details for that configuration. This initial configuration has four video inputs, each with two audio tracks.

Looking further down within the panel, you will see the encoding settings and other details that will be used, as shown in the two columns on the left here:

Option	**Setting**	**Comment**
ENCODE1_BITC	false	true = add burnt-in timecode with encode1
ENCODE1_COPY_DEST	/store/mxf_online/	Destination directory for copy command, e.g. directory on remote server
ENCODE1_COPY_PRIORITY	any integer >0	Sets priority for copy; the lowest integer has the highest priority
ENCODE1_DIR	/video/mxf_online/	Local destination directory for encode1 files
ENCODE1_RESOLUTION	MJPEG 2:1	Encoding selected for the encode1 process

ENCODE2_* [5 parameters]	Similar to above	As above but for encode2
ENCODE3_* [5 parameters]	Similar to above	As above but for encode3
QUAD_* [5 parameters]	Similar to above	As above but for quad-split

The encode1/2/3 resolutions that are available are listed in the table below.

Encoding	**Comment**	**Capture buffer**
Uncompressed		UYVY
Uncompressed MXF OP-ATOM		UYVY
DV25 Raw		DV25
DV25 MXF OP-ATOM		DV25
DV25 Quicktime		DV25
DV50 Raw		DV50
DV50 MXF OP-ATOM		DV50
DV50 Quicktime		DV50
DVCPRO-HD Raw		YUV422 (HD)
DVCPRO-HD MXF OP-ATOM		YUV422 (HD)
DVCPRO-HD Quicktime		YUV422 (HD)
MJPEG 2:1 MXF OP-ATOM		YUV422
MJPEG 3:1 MXF OP-ATOM		YUV422
MJPEG 10:1 MXF OP-ATOM		YUV422
MJPEG 10:1m MXF OP-ATOM		YUV422
MJPEG 15:1s MXF OP-ATOM		YUV422
MJPEG 20:1 MXF OP-ATOM		YUV422
IMX30 MXF OP-ATOM		YUV422
IMX40 MXF OP-ATOM		YUV422
IMX50 MXF OP-ATOM		YUV422
IMX30 MXF OP-1A		YUV422
IMX40 MXF OP-1A		YUV422
IMX50 MXF OP-1A		YUV422
VC3-120/145i MXF OP-ATOM		YUV422 (HD)
VC3-185/220i MXF OP-ATOM		YUV422 (HD)
VC3-36/45p MXF OP-ATOM		YUV422 (HD)
VC3-120/145p MXF OP-ATOM		YUV422 (HD)
VC3-185/220p MXF OP-ATOM		YUV422 (HD)
VC3-120/145i MOV		YUV422 (HD)

VC3-185/220i MOV		YUV422 (HD)
VC3-36/45p MOV		YUV422 (HD)
VC3-120/145p MOV		YUV422 (HD)
VC3-185/220p MOV		YUV422 (HD)
MPEG2 422 Long GOP 50 Mbit/s Raw		YUV422 (HD)
MPEG2 422 Long GOP 50 Mbit/s Quicktime		YUV422 (HD)
MPEG2 for DVD		MPEG
MPEG4/MP3 Quicktime		MPEG
MPEG4/PCM Quicktime		MPEG
SD H264 Baseline Profile 512 kbit/s AAC MOV		MPEG
SD H264 Main Profile 1024 kbit/s AAC MOV		MPEG
SD H264 Baseline Profile 512 kbit/s AAC MP4		MPEG
SD H264 Main Profile 1024 kbit/s AAC MP4		MPEG
SD H264 2 Mbit/s and AAC		MPEG
MP3 Audio only		any
Vision Cuts	For RouterRecorder	n/a

In particular, you probably want to check these settings in the recorder's configuration panel:

- encode formats
- BITC on/off for each encode

To change any of the settings, click on the button "Edit" at the top of the panel, just below the bold text "Ingex – Ingex-MXF" (**not** the Edit button at the top of the page).

After making your changes, click on the button "Done" at the bottom on the panel.

There are many other settings that can be made on the web pages but these are the essential ones at this stage. Further documentation will follow on this in due course.

Capture buffers

If you select an encoding format that requires a different video buffering arrangement, e.g. 4:2:0, you will need to set this on the command line for dvs_sdi. (In time we intend that this will be automatic, but you need to edit the file for now).

To do this, if using the startIngex.sh script, edit the file /etc/ingex.conf (or ~/ingex.conf) and set the required format. For example, for DV25 encoding, set the SECONDARY_BUFFER value to "DV25":

```
# ***** SD Capture options, if required ******
SD_CAPTURE_CHANNELS=4
SD_CAPTURE_MODE="PAL"
SD_CAPTURE_PRIMARY_BUFFER="YUV422"
SD_CAPTURE_SECONDARY_BUFFER="DV25"
```

```
SD_CAPTURE_TIMECODE="LTC"
SD_CAPTURE_OPTIONS=""
```

By default, the video is assumed to be 16x9 aspect-ratio. If you are working 4x3, add the SD capture option “-4x3”.

If you do not need the secondary buffer, you can save some CPU cycles by setting SECONDARY_BUFFER = “None”.

dvs_sdi also has many options and video modes which aren't (yet) described here. To see them all type ./dvs_sdi -h at a command prompt.

Configuring the Controller

Shuttle Pro

The controller and stand-alone player are compatible with a Contour Design Shuttle Pro or Shuttle Pro 2 USB jog/shuttle control. In order for this to function, non-root users need to be able to access the device. This is achieved by unplugging the jog/shuttle control and installing a udev rules file, as follows:

```
ingex> cd $workspace/player/jogshuttle
ingex> sudo cp 51-shuttle-pro.rules /etc/udev/rules.d
```

When you plug the jog/shuttle control back in, it should be available to the controller.

Starting the controller

When it comes to run the controller you will need to use a command like this, but with the IP address of your Nameserver instead of the 192.168.1.181 here:

```
ingex> ingexgui -ORBDefaultInitRef corbaloc:iiop:192.168.1.181:8888
-ORBDottedDecimalAddresses 1
```

(Note that for the controller to work correctly, the Nameserver must be running first, but we'll come to that later.)

The file $workspace/studio/scripts/run_ingexgui.sh runs this command and so it is best to edit this file to use the IP address of your nameserver, and then create a shortcut on the desktop so it can be run easily.

Replaying files

After a recording has finished, the controller will try to load it into the player so that the files are ready immediately for replay, if required. In this process, the filenames given to the controller have the recorder's hostname prepended to the path, to make it easier when multiple recorders are being used. As an example, if a recorder is running on host "ingexRecorder", the controller will be looking for the video files beneath the point /ingexRecorder. You will need to have the recorder's video directory at this point otherwise file replay will not work.

If you are running the controller on the same PC as the recorder, you can achieve this by simply having a symbolic link as follows:

```
ingex> sudo mkdir /<hostname of recorder>
ingex> sudo ln -s /video /<hostname of recorder>/video
```

where /video is the top level directory containing the recordings and <hostname of recorder> is the PC's hostname.

If the recorder is on a separate PC from the controller, you will need to export the recorder's /video directory and mount it on the controller PC. YaST lets you do the export (under Samba Server), then edit the file /etc/fstab on the controller PC, to mount the filesystem locally. For example, add a line like this to /etc/fstab

```
//192.168.1.181/video /ingexhost/video cifs noauto,...etc.,
```

(the options after noauto will depend on your installation).
In this example 192.168.1.181 is the IP address of the recorder (although you could use its hostname instead at that point), and its hostname is ingexhost.

On the controller PC, you can now create the mount point and mount the remote file system with:

```
ingex> sudo mkdir -p /ingexRecorder/video
ingex> sudo mount /ingexRecorder/video
```

Replay via the SDI output

If you want the controller to replay recordings via the SDI output of one of the DVS cards being used for capture, then you must run the controller on the same PC as the recorder, so that it has access to the card.

If you need to control the recorder from a remote PC (i.e. from a studio control room), the easiest way to achieve this is by using a X-session from that PC, to run the Ingex controller application on the recorder itself. In that way the controller has access to the DVS cards, but displays its windows on the remote PC, as required.

You can achieve this with a command on the remote PC such as:

```
ssh -X -c arcfour <hostname> ap-workspace/ingex/studio/scripts/run_ingexgui.sh
```

It's convenient to set up this command as a shortcut on the desktop (and appropriate configuration of ssh means you can avoid it prompting for a password each time it runs).

Transfer Server

The transfer server looks after the copying of recordings from the local discs on the Ingex recorder to a second location, for example, a file server machine or a portable drive plugged into the recorder PC. This means that a second copy of the content is made as soon as possible after a recording has ended, and also makes it possible to deliver the recordings automatically to an edit suite, or other centralised location. It also deletes the oldest recordings from the local discs as they start to become full (and provided they have been copied).

This script is run on each recorder PC and the source and destination directories it uses for the copying are determined by the recorder configuration. It maintains its own configuration file, defined by the constant PATHS_FILE, which allows it to resume interrupted copying when it is started and to avoid copying material repeatedly if the files are removed from the destination directory.

Prerequisites

Perl modules:

- IPC::ShareLite
- Filesys::DfPortable
- IO::Socket
- IO::Select
- IO::File
- Getopt::Std
- Storable
- Term::ANSIColor

Some of these will already be installed: the way to find out what is missing is to run xferserver.pl and see if it aborts, reporting the first package it fails to find. See instructions in the Installation Prerequisites section for how to install Perl modules. If you have already installed them as part of a full system build, you will not need to do it again.

Building the software

To build the software, move to its directory:

```
ingex> cd $workspace/studio/processing/media_transfer
```

and build the executable cpfs, which performs the copying:

```
ingex> make
```

You will see that two Perl scripts are in this directory: xferserver.pl and xferclient.pl.

The location of the executable cpfs that you have just built needs to be defined in the file xferserver.pl if you are moving it from the current directory. Open it in an editor and find the line:

```
use constant COPY => './cpfs';
```

and change **./cpfs** to the location of cpfs on your system.

Whilst you are editing xferserver.pl you may want to change the value for KEEP_PERIOD from the default week to whatever you require and DELETE_THRESHOLD from the default 90 (%). This is the length of time files are left on the Ingex recorder after being copied successfully to the destination, before the local file is deleted, if the disks are full to the capacity given. (These options are to avoid the discs filling too much.)

xferserver.pl has many options. To find out what they are:

```
ingex> ./xferserver.pl --help
```

Director's Cut

The Ingex player software can be used in a mode which records cuts made between the video inputs displayed in the quad split presentation. This is the “Director's Cut” facility and allows the video recordings to be made without interruption, but the cuts between them, viewed live in the quad split, are carried through into an AAF file. When this file is opened in a video editor, the video sequence includes the cut which can then be edited if necessary.

The player already includes this functionality so no further installation is required. The usage details for the player will give the command arguments required, but here is an example, which records the cut details into the file /theo/directors_cut.db:

```
ingex>/usr/local/bin/player --show-tc LTC.0 --audio-lineup -18 --audio-dev 1 --audio-mon 2
--source-aspect 16:9 --quad-split --hide-progress-bar --shm-in 0p --disable-audio --shm-in 1p --shm-
in 2p --shm-in 3p --split-select --vswitch-db /theo/directors_cut.db --vswitch-tc LTC.0
```

It is best to put the command into a script which will ensure that the player starts correctly, if one is already running. See the shell script $workspace/studio/scripts/directors-quad.sh for an example of how to do this.

To make use of the Director's Cut data in the AAF file that is later created,you will need to set the configuration details in the web-page setup. Refer to the details on the Web Server installation and in particular the section on editing the material.conf file. In that file, find the section:

```
# the directors cut database files
# format is <name>|<filename>,<name>|<filename> etc.
directors_cut_db = player|/srv/ingex/directors_cut.db
```

and change the <name>/<filename> pairs on the line 'directors_cut_db' as necessary for your system. This will mean ensuring that the file referred to in the line is the same file as that in the --vswitch-db option shown above in the player command (in that example, /theo/directors_cut.db). You need to make sure that these both refer to the same file. Where the web server is not running on the same PC as the recorder, you will need to arrange access to the file via a network drive, probably from the recorder.

Multicasting the video

You can set up multicasting from the Ingex recorder PC, so that the incoming video feeds can be viewed over the network using remote players. The streams are MPEG compressed and presented as a standard transport stream, so can be viewed on players such as vlc, mplayer and ffplay.

Routing details

First you will also need to add some routing details in YaST, as follows:
YaST->Network Devices->NetworkSettings->Routing->Add
    Destination=224.0.0.0 Gateway=0.0.0.0 Netmask=240.0.0.0

Make sure you select the correct network adaptor, e.g. eth0, - you can confirm the mac address with the command /sbin/ifconfig.

Once the YaST steps are run, you can check the route is in place by running this command and looking for the 224.0.0.0 route:

```
ingex> /sbin/route -n
```

Starting the multicast streams

First move to the capture directory:

```
ingex> cd $workspace/studio/capture
```

Each video channel is multicast separately, using a command like this:

```
ingex> ./nexus_multicast -c 0 -t -q 239.255.1.1:2000 &
```

By default this will produce a stream at 3.5 Mb/s with image size 720x576 pixels. You can experiment with the -b and -s options if you want something different. For example,

```
ingex> ./nexus_multicast -c 0 -t -q -s 360x288 -b 1000 239.255.1.1:2000 &
```

will produce a ¼ picture at 1 Mb/s.

Further streams can be started by setting the source video channel with the -c option and selecting a different port for the outgoing stream, e.g. to multicast the second video input via port 2001, use this command:

```
ingex> ./nexus_multicast -c 1 -t -q 239.255.1.1:2001 &
```

It is easiest to create a script to start the multicast streams as required for your installation. See the file $workspace/studio/scripts/start_multicast.sh as an example – but note that this will need editing for your configuration.

Viewing the multicast streams

To view a multicast stream, use one of these players, with the following udp address formats. Note: The port number (:2000) will need to be incremented for each stream, resulting in four players being open to view four streams etc.

vlc	udp://@239.255.1.1:2000

mplayer	udp://239.255.1.1:2000
ffplay	udp://239.255.1.1:2000

[Note: Using vlc on an ingex recorder is not recommended because of a likely conflict between ffmpeg versions.]

Viewing the uncompressed stream

If you do not use the -t option in the *nexus_mulitcast* command above, an uncompressed but scaled-down picture will be multicast. Note that the data rate will be about 14 Mb/s for each standard definition source.

It's probably better to use the compressed streams described above, but if you wish to avoid compression, you can use this stream and view it using the Ingex player.

A typical command to produce the stream is:

```
ingex> ./nexus_multicast -c 0 -q -s 240x192 239.255.1.1:2000 &
```

Once the player is built, use the command below to start the player and display the streams. Note that in this example, four streams are expected (the -udp-in argument is used four times), so adjust as necessary to match the number of channels that you are streaming. If you try to receive a non-existent stream, the player will wait for it to appear.

```
ingex> player --pixel-aspect 1:1 --quad-split --prescaled-split --hide-progress-bar --source-aspect
16:9 --udp-in 239.255.1.1:2000 --udp-in 239.255.1.1:2001 --udp-in 239.255.1.1:2002 --udp-in
239.255.1.1:2003
```

Troubleshooting

If you are having difficulties getting the multicast streams to work, you can test you network setup by sending a data stream from your Ingex recorder as follows:

```
ingex> cd $workspace/common/tools
ingex> ./send_video
```

Then, on the PCs where you want to receive the stream:

```
ingex> cd $workspace/common/tools
ingex> ./receive_video
```

This will indicate whether the connection has been established, together with packet loss, etc.

Testing without DVS capture cards

dvs_dummy

If you do not have the DVS SDK or video capture cards available, the program “dvs_dummy” provides the easiest way of testing the system. It can be run instead of the program “dvs_sdi” and uses the same command line options, but rather than capturing video, it generates colour bars with timecode.

“dvs_dummy” will have been created in the directory $workspace/studio/capture during the build process.

To run it, move to the capture directory:

```
ingex> cd $workspace/studio/capture
```

and use a command such as this, which will create two dummy inputs:

```
ingex> ./dvs_dummy -c 2 -m 250
```

Changing the argument -c 2 to -c 4 will produce four dummy video inputs.

Note: this runs instead of dvs_sdi, so do not run dvs_sdi at the same time.

The remainder of the Ingex system, e.g. recorder, multicast, etc., can now be run in the normal way.

If you want to capture dummy HD frames, first set the DVSDUMMY_PARAM environment variable, e.g.

```
ingex> export DVSDUMMY_PARAM=video=1920x1080
ingex> ./dvs_dummy -c 2 -m 250
```

testgen

As “dvs_dummy” emulates the DVS SDK and capture cards, using it as described above is the preferred method for testing a system if capture cards (or video signals) aren't available. However, a second program “testgen” can also be used, although the only reason for doing this would be if you want to replay a stored video clip rather than using internally generated colour bars.

You will need to build testgen, as follows:

```
ingex> cd $workspace/studio/capture
ingex> make testgen
```

To replay a stored clip, instead of running dvs_dummy, use this command:

```
ingex> ./testgen -c 4 video.uyvy
```

You can include an audio file too:

```
ingex> ./testgen -c 4 video.uyvy audio.wav
```

To create a suitable video file in uyvy format from an mpeg file, you can use this ffmpeg command line:

```
ingex> ffmpeg -i video-in.mpg -pix_fmt uyvy422 -f rawvideo video-out.uyvy
```

Running the system

The following services should now be run automatically at boot time. These can be checked by looking in *YaST* -> *System* -> *System Services* and checking that they are present and enabled.

- dvs_setup
- Naming_Service
- postgresql
- Apache2

Desktop startup scripts

The easiest way to run up Ingex on KDE desktop is to use the scripts described in the next section “Starting Ingex from the Desktop”.

Below are details for running the components separately without picking up configuration from the ingex.conf file.

Start capture

```
Move to the capture directory
ingex> cd $workspace/studio/capture
```

Start the capture process:

```
ingex> ./capture.sh
```

As the capture process starts, you will see these messages. If you do not have video connected, this will be reported and if there is a problem with timecode, the screen will fill with messages indicating the timecode discontinuities. Once you correct the timecode feed, the messages will stop.

```
05 09:47:13.510554 Capturing video as YUV 4:2:2 planar
05 09:47:13.712388 Secondary video buffer is YUV 4:2:0 planar with picture
shifted down by one line
05 09:47:13.712405 Master timecode type is LTC using channel 0
05 09:47:13.712417 Using LTC to determine number of frames to recover when
video re-aquired
driver version = (3,2,1,0)
05 09:47:13.727347 card 0: device present (720x576)
05 09:47:13.727404 card 0: opened second channel (720x576)
05 09:47:13.727456 card 1: device present (720x576)
05 09:47:13.727512 card 1: opened second channel (720x576)
shmmax=4294967295 (4096.000MiB) probably too big, reducing to 1024MiB
shmmax=1073741824 (1024.000MiB) calculated per channel ring_len=175
element_size=1490432 ring_len=175 (7.00 secs) (total=260825600)
05 09:47:13.730454 chan 0: starting capture thread
05 09:47:13.730503 chan 1: starting capture thread
05 09:47:13.730533 chan 2: starting capture thread
05 09:47:13.730558 chan 3: starting capture thread
05 09:47:13.963867 chan 0: Video signal OK
05 09:47:13.974401 chan 1: Video signal OK
05 09:47:13.989859 chan 2: Video signal OK
05 09:47:13.995984 chan 3: Video signal OK
```

Start Quad-split

If you wish, you can now check the incoming video by starting the player in quad split mode, as follows.

Move to the scripts directory:

```
ingex> cd $workspace/studio/scripts
```

Start the player in quad-split mode:

```
ingex> ./quad-split.sh
```

Note: if your hardware does not have four video inputs, you will need to modify the quad-split.sh script accordingly. (Reduce the number of –shm-in arguments to match the number of video inputs you have, e.g. for a two-channel system, you only need --shm-in 0p --shm-in 1p in the command line.)

Start recorder

Move to the recorder directory

```
ingex> cd $workspace/studio/ace-tao/Recorder
```

Look in the file run_recorder.sh; it will contain a recorder command line similar to this:

```
./Recorder Ingex bamzooki bamzooki -ORBDefaultInitRef
corbaloc:iiop:192.168.1.181:8888 -ORBDottedDecimalAddresses 1 $*
```

Ensure that the corbaloc argument has the IP address of your Naming Service. Here it's 192.168.1.181, but it may be blank in the original version of this file, so put in the appropriate IP address for your configuration.

Start the recorder by running the script

```
ingex> ./run_recorder.sh
```

As the recorder starts, it will display the following messages:

```
Main thread

Recorder name "Ingex"

UpdateSources() loaded config "Default" of recorder "Ingex".
corbaloc:iiop:1.0@192.168.1.181:8888/NameService - object advertised
Recorder running...
```

Ignore the error about not being able to connect to port 2000. That's a warning because the background copy process xferserver.pl isn't running at the moment.

Troubleshooting

If you see a message like this when you start the recorder:

```
Main thread

Recorder name "Ingex"
```

```
UpdateSources() loaded config "10:1m and 2:1" of recorder "Ingex".
corbaloc:iiop:1.0@192.168.1.181:8888/NameService - object could not be
advertised
Pausing before re-attempting
```

it means that the recorder is unable to contact the Naming Service.

Check that:

a) the Naming Service is running. Look in *YaST->System->System Services,* and to see if the entry *tao-cosnaming* is running. If it's not, start it now and also set it to start automatically on power-up. If it's not present in the list, check the sub-section “Installing the ACE and TAO packages” in the earlier section “Installation Prerequisites”.

b) the IP address in the script run_recorder.sh corresponds to the PC running the Naming Service.

c) you don't have a firewall enabled between the PCs.

Start Controller

On the controller PC, move to the directory containing the script that you created when installing the controller, and run it.

For example:

```
ingex> cd $workspace/studio/scripts
ingex> ./run_ingexgui.sh
```

If the recorder is running correctly from the earlier steps, then its name will appear in the “Recorders” list. Click it to select, and timecode display should then start incrementing.

This confirms you have control of the recorder.

Making a recording

In the controller:

- Select the Ingex recorder by clicking on its name in the list at the top

The timecode should be running in the controller display. If it's not, go to recorder configuration web page and check the timecode mode is correct.

- Set tape IDs

Click on the “Set tape IDs” button (upper right), and in one of the columns of the grid, enter a sequence of different numbers. At this stage it doesn't matter what numbers are.

- Press record

It should turn bright red, indicating that the recorder has gone into record.

Checks:

a) look in the window where you started the capture process – are there any error messages? It shouldn't have changed since you started the capture process itself (unless you interrupted the video or timecode).

b) Look in the window where you started the recorder. You will see messages confirming that the recording has started and the encoding formats being used, but you shouldn't see “Frames waiting” messages (although they're not fatal) or, even worse, “Dropped frames” (which are fatal).

- Press stop

If you have followed the controller configuration section, the recordings just made will appear. If they don't, check the setting for the player under “ Player” -> “Player Type” and make sure “Computer screen (accelerated if possible)” is selected.

If this seems correct, then run the controller from a shell prompt and look at the messages that are produced. Initially file not found messages will appear immediately after each recording, but once the files become available, the controller should pick them up for display.

The user documentation covering IngexGUI describes in much more detail how you control Ingex, but we just want to see if it's working at this stage.

- Check the recordings

They should have appeared in the appropriate encode directories (initially /video/mxf_online/ and /video/mxf_offline/). Copy them over to your preferred editing system and check the content. Success - hopefully!

Troubleshooting

1. If the recorder drops frames, it may be because you don't have enough CPU power to run two encodings of each video input simultaneously. As a test, go into the recorder configuration web page and turn off one of the encodings by setting either ENCODE1_RESOLUTION or ENCODE2_RESOLUTION to “Not Set”.

2. If it still drops frames, reduce the number of video channels by using the -c option on the dvs_sdi command line in the capture.sh script.

3. Use 'top' or 'gkrellm' to see how much load is on the system. With two quad-core cpus running at 3 GHz, four SD video inputs, with two encodings each, should load the cores to about 50-60%, on average

Starting Ingex from the Desktop

To make it easier to start the various parts of the Ingex system, you can use the bash script startIngex_new.sh (in $workspace/studio/scripts/) to start and stop the required modules, as set beforehand in the file /etc/ingex.conf. This gives a convenient way for users to run the system after power-up, without having to enter the separate commands.

We'll probably expand the use of /etc/ingex.conf but, at present, its contents includes:

```
# Set the following variables to 1 to enable each facility on startup. Zero
means it won't be started.
CAPTURE=1
HD_MODE=0
MULTICAST=0
TRANSFER=0
INGEX_MONITOR=0
SYSTEM_MONITOR=0
QUAD_SPLIT=1

# CORBA naming service.  Used by IngexGUI to find Recorders.
NAMESERVER="192.168.1.231:8888"

# HD Recorders (space-separated)
HD_RECORDERS="Ingex-HD"

# SD Recorders (space-separated)
SD_RECORDERS="Ingex"

Run the script with the -h option to get the HD configuration.
```

When the bash script $workspace/studio/scripts/startIngex.sh is run, it will open two terminal windows. In the first, a tab is opened for each of the processes in the upper part of the list which is set to 1. In the second, a tab is opened for each recorder named in the “RECORDERS” variable. In this example, there's only one, but it could be a space-separated list.

Initially you probably don't want to run more than capture, a recorder and possibly the quad-split, in this way, so most lines in this file are already set appropriately.

You will need to change the details for the name server, i.e. change the line:

```
NAMESERVER="192.168.1.231:8888"
```

to use the correct IP address of hostname for your system.

You should also check that the name of the recorder in the line:

```
SD_RECORDERS="Ingex"
```

matches the name of your recorder (if you've changed it from the default “Ingex”).

Starting Ingex

The startIngex.sh script will look for the file ingex.conf firstly in your home directory, and then in /etc/. This means you can test new setting and then move the ingex.conf file to /etc when it's ready.

To test the settings, type:

```
ingex> ./startIngex_new.sh
```

Or for HD, use:

```
ingex> ./startIngex_new.sh -h
```

There will be various pauses, but the two terminal windows will eventually open with the necessary tabs. Check that the required processes are running in them.

You can now set a shortcut on the desktop to run the startIngex.sh script so that you can start Ingex with one click.

Stopping Ingex

The same script can be used to end the Ingex processes and close the terminal windows that were created.

The command is

```
ingex> ./startIngex_new.sh -a stop
```

Again, having a desktop shortcut to run this gives an easy way for the user to shutdown Ingex.

Troubleshooting

1) If the new terminal windows open but the processes aren't running within them, then make sure that you do not have more that one or two terminal windows open initially. If there are too many open initially, the script can't keep track of them correctly.

2) If the stopIngex.sh script doesn't close the terminal windows, then check that you can write to /tmp as this is where it stores the PIDs of the relevant processes.

HD Operation

To operate Ingex with HD inputs, you need to set the DVS capture cards in the appropriate mode and then select the required encoding format in the recorder configuration web page.

The configuration of the capture cards is carried out by dvs_sdi, and is set with a command-line option. For example, to operate in “1080i25” mode, the dvs_sdi command line should be:

sudo nice --10 ./dvs_sdi -c 2 -mode 1920x1080i25 -mc 0 -tt LTC -f YUV422

Here the -c 2 option limits the capture to a maximum of two channels, and the -mode option sets the video mode to 1920x1080i25.

To start Ingex in this way, you can use the -h option when you start the capture process with capture.sh. For example:

```
ingex> cd $workspace/studio/capture
ingex> ./capture.sh -h
```

You should see the two channels start in 1920x1080 mode, rather than 720x576.

Refer to the section *Starting Ingex from the Desktop* for details of how you can set the configuration so that Ingex always starts in this mode.

Finally go to the recorder web page and set the HD encoding format that you require.

Starting additional facilities

Start monitoring

Optionally, you can also monitor the operation of the Ingex system from a web page. To do this you need to start the monitoring software on each PC for which this is required.

Move to the directory containing the capture software

```
ingex> cd $workspace/studio/capture
```

Start the process that monitors the Ingex capture and recorder operation:

```
ingex> ./nexus_web
```

You can also monitor system information about the PC too, by starting this process, in another shell:

```
ingex> ./system_info_web
```

This PC can now be selected for monitoring by selecting the 'Status' tab on the Ingex web page at http://<your Ingex web server>/ingex

Start transfer server

This stage is optional and only required if you wish to use the background transfer facility to copy recordings to a second destination, e.g. a remote file server.

On the recorder PC run the xferserver script:

```
ingex> cd $workspace/studio/processing/media_transfer
ingex> ./xferserver.pl
```

In the recorder configuration (set via the web page, as above), you will need to set the entry "Copy Command" for the recorder to be:

```
$workspace/studio/processing/media_transfer/xferclient.pl
```

Then, when a recording has ended, the xferclient.pl script will be called to initiate the copy.

Starting and Stopping Multicasting

To start multicasting streams from the recorder, move to the relevant directory:

```
ingex> cd $workspace/studio/capture/nexus_multicast
```

and start the multicast using the script previously created during your setup of the recorder, i.e.

```
ingex> ./start_multicast.sh
```

Starting and Stopping the Router Logger

It is possible to directly connect a vision mixer to an Ingex recorder via a serial link or over a network to record the director's cut decisions throughout a shoot. The logger can be enabled in the ingex.conf file, but must be configured in the run_routerlogger.sh script. Move to:

```
ingex> cd $workspace/studio/ace-tao/routerlog
```

Open run_routerlogger.sh for editing:

```
ingex> vim run_routerlogger.sh
```

If connecting over a serial cable locate the following text near the top of the file:

```
if[ "$LOCATION" = "Studio"
      then
```

and enter the following below this, filling in the gaps as appropriate for your system to match the router recorder setting configured in Ingex Web:

```
./routerlogger -v -r /dev/ttyS0 -l \
-n <NAME OF ROUTER IN RECORDER SETUP ON INGEX WEB> \
-m <ROUTER O/P DESTINATION> \
-d "<NAME OF ROUTER CHANNEL 1>" -p <CHANNEL NUMBER> \
-d "<NAME OF ROUTER CHANNEL 2>" -p <CHANNEL NUMBER> \
<INSERT AS MANY AS NECESSARY>

-c "<MULTICAM CLIP DEF NAME>" \
-a <NAME SERVICE>
```

For example:

```
./routerlogger -v -r /dev/ttyS0 -l \
-n Ingex-Router \
-m 9 \
-d "VT-1" -p 1 \
-d "VT-2" -p 2 \
-d "VT-3" -p 3 \
-d "VT-4" -p 4 \

-c "Studio-A Multicam" \
-a corbaloc::iiop:192.168.1.181:8888/NameService \
```

If connecting over a network, the procedure is the same with a change to the first line of the script:

```
./routerlogger -v -r <IP ADDRESS OF ROUTER SIGNAL>:4098 -s -l \

eg:
./routerlogger -v -r 192.168.1.238:4098 -s -l \
```

Samba shares for Avid editor and player access

Samba shares can be set up so that an Avid system can access the MXF files directly. This can be done manually or by installing a script as a system service.

Note for opensuse 11.4

The AppArmor profile for samba is broken. You should use Yast -> Novell AppArmor -> Edit Profile to update or delete it.

Manual creation

1. Create a structure such as /exports/avid_online/Avid Mediafiles/MXF
   In the MXF directory, make a suitable symbolic link such as HOST.1 -> /video/mxf_online
   where HOST is the host (Avid client) you want to index the files.

2. Install samba and samba-client using YaST (it also installs another two packages as dependencies).

3. In YaST, Network Services-> Samba Server
a) enable at boot up
b) Add /exports/avid_online as an exported filesystem, with name avid_online
c) Do something similar for avid_offline and avid_aaf
d) Export /video as readonly - this is needed by the controller for replay

4. Edit /etc/samba/smb.conf
With some versions of samba, you need to enable following of symbolic links which go outside the directory that is shared. In the [global] section of smb.conf add the following options:

```
        wide links = Yes
        unix extensions = No
```

5. Add samba user ingex:

```
ingex> sudo su
root> smbpasswd -a ingex
root> exit
```

6. Check samba export is working, by looking at its status:

```
ingex> smbstatus
```

Automatic creation

Introduction

There is a script which will automatically create Samba shares as material is copied to the server. These shares are named <project name>_online or <project name>_offline and should be mounted on the PCs used for editing. They contain directories named Avid MediaFiles/MXF, as required by Avid editing software. Each of these directories contains subdirectories named INGEX.<date> which in turn contain the online or offline material for the corresponding project for that particular date (in the YYYYMMDD format). "INGEX" should be changed manually to the host name of the Avid client which will be editing that material. This will not affect the organisation of the material on the server, because the subdirectories are in fact symbolic links to the existing material directories. (Shares and links are also automatically removed if the corresponding material is deleted from the server. This process is not affected by renaming the subdirectories either.)

Prerequisites

The script to handle these operations is $workspace/studio/processing/media_transfer/export_for_avidd.pl, which is installed as a daemon on a file server. It expects to find the material in the directory tree /<top level>/<material type>/<project name>/<date>/, where

- <top level> is /store (defined as $MATERIAL_ROOT in the script)
- <material type> is mxf_online or mxf_offline (defined as the keys to %VIDEO_DIRS)
- <project name> can be anything
- <date> is 8 digits, as generated by xferserver.pl when copying material from recorders

If Ingex has been installed as recommended in this document, the material should be organised as expected.

Installation

The installation procedure is as follows:

1. Follow steps 2, 3a, 3d, 5 and 6 from the previous section, to install and activate Samba and create the manual share.

2. Make sure that the directories /store/mxf_online and /store/mxf_offline exist, and that the Ingex user has write permission to /store and the two subdirectories. (The created directories and links will be given the same ownership and permissions as /store.)

3. Install the required Perl modules for import_db_infod.pl if you don't have them:

```
Proc::Daemon
Linux::Inotify2
```

See instructions in the Installation Prerequisites section for how to install Perl modules. If you have already installed them as part of a full system build, you will not need to do it again.

4. Test the script by running it in non-daemon mode:

```
ingex> cd $workspace/studio/processing/media_transfer
ingex> sudo ./export_for_avidd.pl -n
```

The script should report that it is scanning directories and creating shares and links if there is any material. (These messages are normally just sent to the system log.) If you make a recording with a new project name, when it is copied to the server the script should respond accordingly by generating a new share and link. You should be able to mount the created Samba shares on a Windows PC. Press Ctrl-C to stop the script.

5. Install as a system service:

```
ingex> sudo cp export_for_avidd.pl /usr/local/sbin
ingex> sudo cp export_for_avidd /etc/init.d
ingex> sudo /sbin/chkconfig -a export_for_avidd
```

When run, the script scans the <top level>/<material type> directories for existing <project name> subdirectories and then watches these directories for subsequent creation or deletion of <project name> subdirectories. It maintains a corresponding set of Samba export points at /store/samba-exports/ (defined as $SHARES_ROOT and $SHARES_DIR respectively in the script). These exports are from subdirectories called <project name><type> where <type> is _online or _offline depending on the <material type> (defined as the values of %VIDEO_DIRS in the script). Their names are the same as the subdirectories. So, for instance, if there is a directory /store/mxf_online/MyProject/ there will be a share named MyProject_online.

Additional file server database

If recordings are being copied to a file server and recorders are using database(s) on recording machines rather than this file server, an additional slave database can be set up on the file server to store details of the recordings as they are copied across. This gives the recording and post-production areas independence from each other, allows queries for material browsing etc. to be diverted from critical recorder databases and aggregates all the recording details if multiple recorder databases are in use.

Assuming you have set up transfer servers on each recording machine to copy to your file server (see p37), and have a database on your file server (see p23), the following steps need to be taken on the file server to enable automatic updating of the file server's database:

1. Build and install import_mxf_info, which extracts recording information from MXF files and puts it in the database:

```
ingex> cd $workspace/studio/mxfreader
ingex> sudo make install
```

2. Build and install import_cuts, which copies router recorder data into the database:

```
ingex> cd $workspace/studio/processing/import_cuts
ingex> sudo make install
```

3. Make sure the root material directory is present:

```
ingex> sudo mkdir -p /store
ingex> sudo chown ingex /store
```

(Note that you will probably be mounting a storage array at this point.)

4. Install the required Perl modules for import_db_infod.pl if you don't have them:

```
Proc::Daemon
Linux::Inotify2
```

See instructions in the Installation Prerequisites section for how to install Perl modules. If you have already installed them as part of a full system build, you will not need to do it again.

5. Test import_db_infod.pl, which detects new files as they are copied onto the server, and calls the above two executables accordingly:

```
ingex> cd $workspace/studio/processing/media_transfer
ingex> ./import_db_infod.pl -n
```

If you already have material in /store, you should get messages about importing files. If you have a recording machine with copying set up, when you make a recording you should see the files being imported.

6. Install import_db_infod as a system service so that it will run whenever the file server is started:

```
ingex> sudo cp import_db_infod.pl /usr/local/sbin
ingex> sudo cp import_db_infod /etc/init.d
ingex> sudo /sbin/chkconfig -a import_db_infod
ingex> sudo /etc/init.d/import_db_infod start
```

You can monitor output from the daemon with:

```
ingex> sudo tail -f /var/log/messages
```

Building the Recorder

This section describes how to build the Ingex recorder software separately from the full build. If you have already built the full suite by following the section 'Building the full system', you can ignore this section.

These notes are derived from the file $workspace/studio/docs/RECORDER.txt, which will be updated first when any changes to the instructions are made.

Prerequisites

If you haven't already, you need to follow the section *Installation Prerequisites* to install the packages, pre-built RPMs, DVS SDK and AAF toolkit. However, not all the standard packages listed there are required by the recorder - you only need to install these from YaST:

libuuid-devel
libjpeg-devel
Xerces-c
libXerces-c-27
libXerces-c-devel

If you selected the software options suggested in "Installing openSUSE" during installation, then the following packages should already be installed. Install them now if YaST indicates they're not present.

kernel sources
libuuid1
fontconfig-devel
freetype2-devel

Building the MXF library and Recorder

Build and install the YUVlib library:

```
ingex> cd $workspace/YUVlib
ingex> make
ingex> sudo make install
```

Build and install the libMXF and libMXF++ libraries:

```
ingex> cd $workspace/libMXF
ingex> make
ingex> sudo make install
ingex> cd $workspace/libMXF++
ingex> make
ingex> sudo make install
```

Build the common files:

```
ingex> cd $workspace/common
ingex> make
```

Some errors that can occur with this are listed here.

Cannot find -lwritearchivemxf	You haven't built and installed libMXF in the step above
libMXF tests cannot be run without MXFDump	Ignore this as it doesn't matter for this build. (Alternatively you can install the AAFSDK (see its README.txt) if you really need to build this.)
DVSSDK not set and could not be detected	You didn't set the DVSSDK environment variable in DVS driver build instructions above, or you're not using the shell where you set it (it's easiest to put the export DVSSDK line in your ~/.bashrc file so it's always set).

Next make the capture, multicast, monitoring and test software:

```
ingex> cd $workspace/studio/capture
ingex> make
```

You should now have the capture program, "dvs_sdi", multicast server "nexus_multicast", monitoring software "nexus_web" and "sys_info_web", and the test software "dvs_dummy" in the above directory.

Note that if you do not have the DVS SDK present, the capture software "dvs_sdi" will not have been built during the last 'make'. In this case you can continue and test the full system by running "dvs_dummy" instead of "dvs_sdi".

Continuing the build, next make the software for handling MXF files.

```
ingex> cd $workspace/studio/mxfwriter
ingex> make
```

Next run the ACE configuration script:

```
ingex> cd $workspace/studio/common
ingex> $ACE_ROOT/bin/mwc.pl -type gnuace
```

Now build the files:

```
ingex> make
```

Now build the recorder:

```
ingex> cd $workspace/studio/ace-tao
ingex> $ACE_ROOT/bin/mwc.pl -type gnuace
```

The previous command will produce the following two errors, but these can be ignored:
CIAO_ROOT was used in the configuration file, but was not defined.
DDS_ROOT was used in the configuration file, but was not defined.

```
ingex> make Recorder-target
ingex> make RecorderClient-target
```

The Ingex recorder is now built and you should now have the executable:
  $workspace/studio/ace-tao/Recorder/Recorder

Building Ingex Player

The player is a versatile piece of software that is extremely useful in the Ingex system. It allows mxf files to be replayed, the incoming video to be displayed during live capture and the multicast streams to be viewed.

This section describes how to build the Ingex player software separately from the full build. If you have already built the full suite by following the section 'Building the full system', you can ignore this section.

These notes are derived from the file $workspace/player/README.txt, which will be updated first when any changes to the instructions are made.

Prerequisites

1. Standard packages

If you haven't already installed these, then open YaST and install:

```
  libuuid1
  libuuid-devel
  portaudio-devel
```

Also required, but are part of the standard SUSE installation, are these

```
  freetype (font library)
  portaudio
```

2. Build and install libraries

If this hasn't already been done, build the YUVlib and libMXF software (you may have done this already as a part of the recorder installation, so it may return immediately, reporting it's up-to-date):

```
ingex> cd $workspace/YUVlib
ingex> make
ingex> sudo make install
ingex> cd $workspace/libMXF
ingex> make
ingex> sudo make install
```

You will also need to make the common library:

```
ingex> cd $workspace/common
ingex> make
```

3. FFmpeg

Note: The player will still compile if you don't have FFmpeg but you will not be able to play files.

If you haven't already installed the FFmpeg rpm then follow the instructions already given in the section "Installation Requisites". You will need to install both the codecs_for_ffmpeg and ffmpeg rpms.

4. DVS SDK

The player will compile successfully if you don't have the DVS SDK, however, you will not be able to replay through the SDI output.

If you haven't already installed the DVS SDK then follow the instructions already given in the section "DVS card drivers and SDK".

Ensure that you have set the DVSSDK environment variable (as explained in the SDK installation steps), so that the build commands below can find the SDK files.

5. Header file nexus-control.h

Normally ingexPlayer will be built within the entire Ingex source tree checked out from CVS and so this note can be ignored. However, if you are building the player separately from rest of the Ingex software, then the header file nexus-control.h is required if replay from shared memory (e.g. the live quad split) and playback through an SDI output is to be included in the build. This file is in $workspace/studio/capture and it's easiest just to checkout that directory too.

Installation

With the prerequisites complete, you can build the player:

```
ingex> cd $workspace/player/ingex_player
ingex> make
```

If you are going to build the Ingex GUI software later, then build the C++ library for the player:

```
ingex> cd $workspace/player/IngexPlayer
ingex> make
```

You can test the player with these commands. The first prints the help details, and the second produces a pattern of moving balls.

```
ingex> ./player -h
ingex> ./player --balls 10
```

If it's working correctly, install the player, with the following command. (You can safely ignore any error message that reports "no DVS available".)

```
ingex> sudo make install
```

Building IngexGUI

Ingex GUI is the controller for the recorders. You can install this on a recorder PC or, where remote operation is required, for example from a studio control room, on a smaller separate PC, sited elsewhere.

This section describes how to build the Ingex GUI software separately from the full build. If you have already built the full suite by following the section 'Building the full system', you can ignore this section.

These notes are derived from the file $workspace/studio/ace-tao/Ingexgui/README, which will be updated first when any changes to the instructions are made.

Prerequisites

1) In YaST, install these packages:
   autoconf,
   automake
   libtool
   wxWidgets-devel

2) Build the C++ version of the player in $workspace/player/IngexPlayer/ - for details, see the section "Building Ingex Player". If the player is built with DVS card support, IngexGUI will be similarly built (and will be able to play back through a DVS card).

3) Install ACE/TAO - see the details in the sub-section "ACE and TAO" in the earlier section "Installation Prerequisites".

4) Build libidl if you haven't already built another CORBA application, such as the recorder, on this PC. (Remember that $workspace in these command represents the path to the Ingex Studio software, so substitute the full path, if you haven't set the environment variable as suggested earlier.)

```
ingex> cd $workspace/studio/ace-tao/IDL
ingex> $ACE_ROOT/bin/mwc.pl -type gnuace
ingex> make
```

(Don't worry about the CIAO_ROOT and DDS_ROOT warnings.)

Building Ingex GUI

If you're not sure, check that $ACE_ROOT and $DVSSDK (if player has DVS card support) are defined, by seeing if these commands report sensible locations:

```
ingex> echo $ACE_ROOT
ingex> echo $DVSSDK
```

Move to the directory containing the controller software:

```
ingex> cd $workspace/studio/ace-tao/Ingexgui
```

Now run these command to build the software:

```
ingex> rm ltmain.sh
ingex> libtoolize
ingex> autoreconf
ingex> ./configure
ingex> make
```

The executable, ingexgui, is built in the sub-directory src.

If the build is successful, install the executable:

```
ingex> sudo make install
```

Troubleshooting

If you see...	**Check...**
Cannot find -ld3mxfinfo or -lmxfreader	You probably didn't build and install the libMXF libraries. Check these in the section on building the Player.