How to stream data between 2 cores over a channel

version	1.0.1
scope	Example. This code is provided as example code for a user to base their code on.
description	How to stream data between 2 cores over a channel
boards	Unless otherwise specified, this example runs on the SliceKIT Core Board, but can easily be run on any XMOS device by using a different XN file.

By default, channel I/O is synchronous. This means that for every byte/word sent over the channel there is some handshaking taking place and also that the task performing the output is blocked until the input task at the other end of the channel has received the data. The time taken performing the synchronization along with any time spent blocked can result in reduced performance. Streaming channels provide a solution to this issue. They establish a permanent route between 2 tasks over which data can be efficiently communicated without synchronization.

To stream data between cores you first need to declare a streaming channel e.g.

```
streaming chan c;
```

You can then pass each end of the channel to each logical core, thus opening a permanent route between the 2 cores.

```
par {
    f1(c);
    f2(c);
}
```

This function outputs the value of 1 on the channel streaming ‘c’

```
void f1(streaming chanend c) {
    c <: 1;
}
```

This function inputs the value of 1 from the streaming channel ‘c’.

```
void f2(streaming chanend c) {
    int i;
    c :> i;
    printintln(i);
}
```

Publication Date: 2013/3/15 REV A