INTRODUCTION TO CHANNELIZATION ALGORITHMS IN SDR AND COMPARISON OF THEM

Presentation by: Mehdi naderi soorki
Instructor: Professor M. J. Omidi
1386-1387 Spring
• the ideal Software Defined Radio (SDR) is all signal processing is done in software, therefore we can support multiple communications channels

• A wideband transceiver has to simultaneously deal with hundreds to few thousands channels

• One of the most computation intensive tasks in such receiver or transmitter is channelization
• subjects forcing the channelization architecture:

1. Spectral Content of the Wideband Channel

   - the system architecture typically defines a fixed carrier spacing with a constant RF bandwidth per carrier channel, such as cellular phone

   - both the carrier frequency and RF bandwidth per carrier are dynamically assigned. This type of architecture is found in a multi-standard communications system such as a multi-standard satellite gateway.
2- **Processor Selection for Channelization Processing:**

General purpose processors allow for maximum reuse of application code across multiple platforms and maximizing the return on investment in application software through code reuse and upgradability but:

- The power utilization and heat dissipation of GPPs are often preventing in many size, weight, and power limited systems. As a result, Digital Signal Processors (DSPs) are often utilized to supplement the processing provided by the GPP to keep the architecture within the specified power budget.

- GPPs and DSPs employ a serial processing architecture that does not provide sufficient performance for the processing of wideband signals. As such, the use of FPGAs, is often required in the SDR platform.

Front-end channelization processing limited to FPGAs due to performance constraints in dealing with the wideband input, although back-end processing which is performed on a per channel basis may incorporate DSPs or GPPs.
channelization is a process where single, few, or all channels from a certain frequency band are separated for further processing. The separation of single channel is usually done by down-conversion followed by filtering and optional sample-rate conversion.

Channelization Algorithms:

1. The per-channel Approach.

2. Pipelined Frequency Transform.

3. Polyphase FFT.
1. The per-channel Approach:
A simple implementation is to use a single-channel channelizer for each channel of interest, and connect them all to the input frequency band signal.
2. Pipelined Frequency Transform:
The Pipelined Frequency Transform (PFT) algorithm is based on a binary tree of DDCs and SRCs.
This algorithm for itself has no advantage over the algorithm presented in the previous section and is actually much more expensive in terms of silicon use, since apart of a single-channel channelizer for each channel of interest in the last stage of the tree, many more are needed in the other stages.

Advantage:

**Smart realization.**

**Half band filters.**

**Half of the sampling rate.**

**Intermediate output.**

**Same action in each step**
3. **Polyphase FFT**:

This channelization algorithm is an improvement of FFT channelization using a polyphase filterbank in combination with FFT.

\[ \text{DDC+LPF} = \text{BPF} + \text{DDC} \]
Algorithms Comparison:

1. Hardware Complexity Comparison:
   
   The first comparison is for LUT (logic use comparison on FPGAs basic block) utilization

   The second comparison is of memory bits utilization.
2. Qualitative Comparison:

**Computational complexity:**
which is usually derived from simulations and software implementations.

**Silicon cost(size)**
Comparison is based on actual implementation in FPGA

**Group delay**

**Flexibility:**
1. Initial design
2. Reconfiguration

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computational Complexity for high number of channels</td>
<td>Per-Channel</td>
</tr>
<tr>
<td></td>
<td>Poor</td>
</tr>
<tr>
<td>Silicon Cost Efficiency</td>
<td>up to 3-20 channels</td>
</tr>
<tr>
<td>Group Delay</td>
<td>Good</td>
</tr>
<tr>
<td>Initial Design Flexibility</td>
<td>Independent channels</td>
</tr>
<tr>
<td>Number of channels</td>
<td>Selectable</td>
</tr>
<tr>
<td>Intermediate outputs</td>
<td>No</td>
</tr>
<tr>
<td>Flexibility for Reconfiguration</td>
<td>Addition / removal of channels</td>
</tr>
<tr>
<td>Filtering independence</td>
<td>Poor</td>
</tr>
</tbody>
</table>
Conclusion:

we introduced three different channelization algorithms. Namely, the per-channel, the PFT, and the PFFT algorithms, explaining in details. then we compare them.

based on HW comparison => PFFT algorithm is better.

Based on qualitative comparison => per-channel algorithm is better.

So the designers must trade off ……
Thank you for attention

Any question?