High Accuracy Wireless Timing Synchronization Using Software Defined Radios

2023 GNU Radio Conference
High Performance SDR Applications

Jason M. Merlo and Jeffrey A. Nanzer
Michigan State University, East Lansing, MI, USA
Outline

1. Motivation and Applications
2. Synchronization Technique
3. Software Overview & Demo
4. Experimental Results
1 | Motivation and Applications
Motivation // Coherent Distributed Antenna Arrays

Traditional Phased Array

- Far-Field Information Wavefront

Distributed Phased Array

- Far-Field Information Wavefront

<table>
<thead>
<tr>
<th></th>
<th>Time Synchronization</th>
<th>Frequency Synchronization</th>
</tr>
</thead>
</table>
Applications of Coherent Distributed Arrays

Next Generation Satellite Cellular Networks

Distributed V2X Sensing

Space Communication and Remote Sensing

Precision Agricultural Sensing

Single-platform resolution

Sensing and communication relay satellite constellation

Distributed array resolution

Earth-Based communication base station

Receiver or imaging target

High Performance SDR Applications
Coherent Distributed Array Synchronization

Node: 1

Wireless Coordination

Node: 2

Destination

Time Synchronization

Phase Alignment

Frequency Syntonization

\[ s_1 + s_2 = \sum_{n=1}^{2} \alpha_n (t - \delta t_n) \exp\{j[2\pi(f + \delta f_n) + \phi_n]\} \]
2 | Synchronization Technique
System Time Model

• Local time at node $n$:
  \[ T_n(t) = t + \delta_n(t) + \nu_n(t) \]
  
  • $t$: true time
  
  • $\delta_n(t)$: time-varying offset from global true time
  
  • $\nu_n(t)$: other zero-mean noise sources
  
  • $\Delta_{0n}(t) = T_0(t) - T_n(t)$

• Goal:
  
  • Estimate and compensate for $\Delta_{0n}$
Time Synchronization Overview

Two-Way Time Synchronization

- **Assumptions:**
  - Link is **reciprocal** ⇒ **quasi-static** during the synchronization epoch

- Timing skew estimate:
  \[ \Delta_{0n} = \frac{(T_{RX0} - T_{TXn}) - (T_{RXn} - T_{TX0})}{2} \]

- Inter-node range estimate:
  \[ D_{0n} = c \cdot \frac{(T_{RX0} - T_{TXn}) + (T_{RXn} - T_{TX0})}{2} \]

For compactness of notation: \( T_m(t_{TXn}) = T_{TXn} \)
High Accuracy Delay Estimation

• The delay accuracy lower bound (CRLB) for time is given by

\[
\text{var}(\hat{t} - \tau) \geq \frac{1}{2\zeta_f^2} \cdot \frac{N_0}{E_S}
\]

• \(\zeta_f^2\): mean-squared bandwidth
• \(N_0\): noise power spectral density
• \(E_S\): signal energy
• \(\frac{E_S}{N_0}\): post-processed SNR

References:

High Accuracy Delay Estimation

\[
\text{var}(\hat{t} - \tau) \geq \frac{1}{2\zeta_f^2} \cdot \frac{N_0}{E_s}
\]

- For constant-SNR, maximizing \(\zeta_f^2\) will yield improved delay estimation

\[
\zeta_f^2 = \int_{-\infty}^{\infty} (2\pi f)^2 |G(f)|^2 df
\]

- \(\zeta_f^{2_{(\text{LFM})}} = (\pi \cdot \text{BW})^2 / 3\)
- \(\zeta_f^{2_{(\text{two-tone})}} = (\pi \cdot \text{BW})^2\)

---

Delay Estimation

• Discrete matched filter (MF) used in initial time delay estimate

\[
s_{\text{MF}}[n] = s_{\text{RX}}[n] \otimes s_{\text{TX}}^*[−n] = \mathcal{F}^{-1}\{S_{\text{RX}}S_{\text{TX}}^*\}
\]

• High SNR typically required to disambiguate correct peak

• Many other waveforms exist which balance accuracy and ambiguity

Delay Estimation Refinement

- MF causes estimator bias due to time discretization limited by sample rate
- Refinement of MF obtained using Quadratic Least Squares (QLS) fitting to find true delay based on three sample points

\[
\hat{t} = \frac{T_s}{2} \frac{s_{\text{MF}}[n_{\text{max}} - 1] - s_{\text{MF}}[n_{\text{max}} + 1]}{s_{\text{MF}}[n_{\text{max}} - 1] - 2s_{\text{MF}}[n_{\text{max}}] + s_{\text{MF}}[n_{\text{max}} + 1]}
\]

where

\[n_{\text{max}} = \arg \max_n \{s_{\text{MF}}[n]\}\]

Delay Estimation Refinement

• QLS results in small residual bias due to an imperfect representation of the underlying MF output

• Residual bias is a function of waveform and sample rate

• Can be easily corrected via lookup table

3 | Software Overview
Software Challenges

1. High/full sample rate with low CPU utilization
   → Use “bursty” transmission scheme

2. Reasonably low latency
   → Use message/PDU-based flowgraph

3. Maintain groupings of PDUs for each channel transmitted/received
   → Use lists of PDUs; initially created a “Wide PDU” type, but switched for compatibility with existing codebase
Software Guiding Principles

• Code reusability
  → Implemented on top of DELTA Python Package for code reusability

• Implementation/iteration speed
  → Scientific processing implemented in Python first, data manipulation in C++
  → Benchmark, re-implement in C++ if necessary
Time Estimation Process

1. **Generate Waveform**
2. **TX Window**: 
   - Matched Filter
   - Peak-Find
   - QLS Estimate
3. **Processing**
4. **RX Window**: 
   - Matched Filter
   - Peak-Find
   - QLS Estimate

**Processing** is shared between nodes to determine corrections.
Time Transfer Flow Graph

USRP Sync Burst

- **Durations:**
  - `time_sync_mf_out`

- **Inputs:**
  - `time_burst_mux`
  - `update_bias`
  - `bias_update`

- **Outputs:**
  - `time_burst_mux`
  - `bias_update`

**Notes:**
- If time sync is currently in progress to prevent other blocks from transmitting during sync epoch, else "1"
Wavegen Block

Virtual Source Stream ID: time_burst_mux
Virtual Source Stream ID: bias_update
Virtual Source Stream ID: mfm_out

Wavegen (py)
Waveform List: wfm..._wo_tone
Log Level: INFO

Vector Operation (py)
Function: [lib... or y in x] style
Log Level: WARN

Virtual Sink Stream ID: mfm_out

Virtual Source Stream ID: sync_burst_rx_out
Virtual Source Stream ID: mf_out

USRP Sync Burst
Samples: 6k
Sample Rate (Sps): 200M
Carrier Frequency List: 2.50E+09, 2.07E+09, 2.50E+09, 2.07E+09
Burst Delay (ms): 5m
Device Arguments: ed.8.11.2
Wire Format: u8c8
Freq. Ref. Source List: internal, external...
PPS Source List: int, ext
TX Stream Channels: 0, 1, 2, 3
TX Gain (dB): 21.0, 10.1, 0.0
TX Ports: TX0, TX1...
RX Stream Channels: 0, 1, 2, 3
RX Gain (dB): 0.0, 0.0, 0.0, 0.0
RX Ports: RX0, RX1...
B-Series Mode: false
Print Debug Messages: false

Outputs dict of WPDUs, keys: 'tx_pdu', 'rx_pdu'
each containing a WPDU of the transmitted and received messages from all channels

Variable
ID: NoiseChannels
Value: [values... for Channels]'

Time Sync Controller (py)
Channels Dict: {0:...}
GPID Dict: {1:...}
Sample Rate: 200M
Waveform Types: Two...WPDUs
Bandwidth: 40M, 40M
Pre-pad: 10n, 10n
Post-pad: 10n, 10n
Rise-time: 5n, 5n
Fall-time: 5n, 5n
Duration: 10n, 10n
Carrier Freq: 2.5G, 2.5G
Ref. Phases: 0.0, 0.0
Label: None
Hist. Len.: 30
Log Level: INFO

sync.en outputs '1' if time sync is currently in progress to prevent
other blocks from transmitting
during sync epoch, else '0'

High Performance SDR Applications
20
USRP Sync Burst Block

USRP Sync Burst

Samples: 61
Sample Rate (Sps): 200M
Carrier Frequency List: 3.328E+09, 2.07E+09, 2.332E+09, 2.302E+09, 2.402E+09, 2.492E+09
Burst Delay (μs): 1μ
Device Arguments: ed.8.11.2
Wire Format: ucl
Freq. Ref. Source List: internal, external
PPS Source List: int. external
TX Stream Channels: 0.1, 2.3
TX Gain (dB): 21.0, 10.0, 10.0
TX Port: RX/TX
RX Stream Channels: 0.1, 2.3
RX Gain (dB): 0.0, 0.0, 0.0
RX Port: RX/TX
B-Serie Mode: false
Print Debug Messages: false

Outputs dict of WPDAs, keys: 'tx_pdu', 'tx_pdu' each containing a WPDU of the transmitted and received messages from all channels.
PDU Matched Filter Block

USRP Sync Burst

Samples: 64
Sample Rate (Spa): 200M
Carrier Frequency List: 1.35GHz, 1.35GHz, 1.35GHz, 1.35GHz, 1.35GHz, 1.35GHz, 1.35GHz, 1.35GHz, 1.35GHz
Burst Delay (ns): 150ns
Device Arguments: "sdr_args addr0=192.168.11.2"
Wire Format: "slice"
Freq. Ref. Source List: "internal,external,rf"
PPS Source List: "int,external"
TX Stream Channels: 4,2,1,0
TX Gain (dB): 21.0,11.0,11.0,11.0
TX Port (TX):-RX:TX:RX
RX Stream Channels: 4,2,1,0
RX Gain (dB): 5.0,5.0,5.0,5.0
RX Port (TX):-RX:TX:RX
B-Series Mode: false
Print Debug Messages: false

Outputs dict of WPDUs, keys: 'tx_pdu', 'rx_pdu' each containing a WPDU of the transmitted and received messages from all channels

High Performance SDR Applications
PDU QLS Peak Estimator Block

USRP Sync Burst

Sampled: 0
Sample Rate (Spa): 200M
Carrier Frequency List: 2.5G, 2.5G, 2.07G, 2.07G, 2.07G, 2.07G, 2.07G, 2.07G...

Burst Delay (s): 1m
Device Arguments: 'd.11.2'
Wire Format: 'utc'
Freq, Ref, Source List: internal, external
PPS Source List: int., Jurnal
TX Stream Channels: 0,1,2,3
TX Gain (dB): 21.0,10.10.0
RX Stream Channels: 0,1,2,3
RX Gain (dB): 0.0,0.0,0.0
RX Port: TX.RX.TX.RX
B-Serie Mode: false
Print Debug Messages: false

Outputs dict of WPDU's, keys: 'tx_pdu', 'rx_pdu' each containing a WPDU of the transmitted and received messages from all channels.

Variable
ID: N0 RX MF Out
Value: N0 RX MF Out

Time Sync Controller (py)
Channels Dict.: '0...2, 3, 3.5'
GPIO Dict.: {'...': 2000}
Sample Rate: 200M
Waveform Types: two-tone, two-tone
Bandwidth: 4MHz, 4MHz
Pre-pad: 10ns, 10ns
Post-pad: 10ns, 10ns
Bias-Delay: 5ns, 5ns
Full-time: 5ns, 5ns
Duration: 10ns, 10ns
Carrier Freq.: 2.5G, 2.07G
Ref. Phase.: 0, 0
Label: none
Hist. Len.: 30
Log Level: INFO

Sync Estimation
ID: N0 RX MF Out
Value: N0 RX MF Out

PDU QLS Peak Est. (py)
Start Value: 20 MHz
Step Size: 5ns
RF Center Freq.: 2.5G
Waveform Types: two-tone
Sample Rate: 200M
Waveform Bandwidth: 4MHz
Correction File Path: Log Level: INFO

Virtual Sink
Stream ID: sync_burst_rx_out

High Performance SDR Applications
Time Sync Controller Block

USRP Sync Burst

Variables:
- ID: NodeChannels
- Value: <class 'dict'>

Time Sync Controller (py) Channels Dict:
- (0, 2, 3)
- Sample Rate: 200M
- Waveform Type: lfm, 2-tone
- Bandwidth: 40M, 40M
- Pre-pad: 10s, 10s
- Post-pad: 10s, 10s
- Rise-time: 5s, 5s
- Fall-time: 5s, 5s
- Duration: 10s, 10s
- Carrier Freq: 1.35G, 2.35G
- Rel. Phase: 0, 0
- Channels: [0, 2, 3]
- Label: N0
- Hist. Len.: 30
- Log Level: WARN

Sync, an outputs ‘1’ if time sync is currently in progress to prevent other blocks from transmitting during sync epoch, else ‘0’

 Outputs dict of WPDOs, keys: ‘tx_pdu’, ‘tx_pdu’ each containing a WPDO of the transmitted and received messages from all channels
Software Demo
Software Demo

Time Transfer
Antennas

Coordination
Server

Radios
(X310s)
High Performance SDR Applications
4 | Experimental Results

Beamforming
Initial PPS Sync

Coarse alignment ~10 ns

TWTT Exchange $\Rightarrow$ Update $\Delta_{0n}$
Residual bias compensated to picosecond level

Transmit beamforming pulses
Compensate using $\Delta_{0n}$, and beamsteer using $\tau_{bf,n}$ and $\phi_{bf,n}$

Estimate $\tilde{\tau}_{bf,n}$ and $\tilde{\phi}_{bf,n}$ at target

Beamsteering

\[ \tau_{bf,n} = \frac{D_n}{c} \sin \theta_{bf} \]
\[ \phi_{bf,n} = 2\pi f_{ci} \tau_{bf,n} \]

First Pulse?

Save $\tau_{bf}$ and $\phi_{bf}$ as calibration

No

Yes

Performance Evaluation Waveforms

• Each node transmitted orthogonal LFM s followed by two CW pulses

Node 0

Node 1

LFM: Time/Phase Estimation

CW Pulses: Coherent Gain Estimation

Experimental Configuration

Transmit Nodes Setup

Target Node Setup (41 m downrange)

Beamforming Results

Induced Frequency Transfer Failure

Beamforming Results

Coherent Gain: 0.91

Nodes 0+1

Coherent Gain: 0.95

Nodes 0+1

Coherent Gain: 0.94

Nodes 0+1

Measurement Summary

Demonstrated fully wireless outdoor time-frequency synchronization and beamforming with $G_c > 0.9$ over a 41 m

<table>
<thead>
<tr>
<th>Internode Distance</th>
<th>Min. Time Transfer Std.</th>
<th>Min. Beamforming Std.</th>
<th>Max. Throughput*</th>
<th>Max. Carrier Frequency†</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 m</td>
<td>10.47 ps</td>
<td>18.00 ps</td>
<td>5.56 Gbps</td>
<td>2.78 GHz</td>
</tr>
<tr>
<td>5.0 m</td>
<td>14.79 ps</td>
<td>24.02 ps</td>
<td>4.16 Gbps</td>
<td>2.08 GHz</td>
</tr>
</tbody>
</table>

* Maximum theoretical BPSK throughput; Pr($G_c \geq 0.9$) > 0.9
† Maximum theoretical carrier frequency; Pr($G_c \geq 0.9$) > 0.9

Project Status and Conclusion

In Progress:
• Standardizing inter-block communications (use PDUs/list of PDUs)
• Complete fully distributed compute software implementation
  • Testing in progress
• Adding/improving documentation

Planned Work:
• Add test cases for CI/CD
• Open source releases
• Investigate use of streaming interface with managed latency to leverage existing streaming blocks
Questions?
merlojas@msu.edu

Thank you to our project sponsors and collaborators:

This work was supported under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DEAC52-07NA27344, by the LLNL-LDRD Program under Project No. 22-ER-035, by the Office of Naval Research under grant #N00014-20-1-2389, and by the National Science Foundation under Grant #1751655.