New Approaches for Direct Current (DC) Balanced SpaceWire
Session: SpaceWire Networks and Protocols, Short Paper

Prepared by:
Alex Kisin, AS&D, Inc. work performed for NASA GSFC
alexander.b.kisin@nasa.gov
and Glenn Rakow, NASA GSFC
glenn.p.rakow@nasa.gov

To be presented by Alex Kisin at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
### List of Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMV</td>
<td>Common Mode Voltage</td>
</tr>
<tr>
<td>DC</td>
<td>Direct Current</td>
</tr>
<tr>
<td>DCF</td>
<td>Data Control Flag</td>
</tr>
<tr>
<td>EEP</td>
<td>Error End of Packet</td>
</tr>
<tr>
<td>ESC</td>
<td>Escape</td>
</tr>
<tr>
<td>EOP</td>
<td>End of Packet</td>
</tr>
<tr>
<td>FCT</td>
<td>Flag Control Character</td>
</tr>
<tr>
<td>FPGA</td>
<td>Field Programmable Gate Array</td>
</tr>
<tr>
<td>LFSR</td>
<td>Linear Feedback Shift Register</td>
</tr>
<tr>
<td>LVDS</td>
<td>Low Voltage Differential Signaling</td>
</tr>
<tr>
<td>PRS</td>
<td>Pseudo-Random Sequence</td>
</tr>
<tr>
<td>XOR</td>
<td>Exclusive OR</td>
</tr>
</tbody>
</table>

4b/5b, 4b/6b, 8b/10b, 8b/20b, 16b/30b – Different Data Encoding Schemes for DC Balancing

---

**New Approaches for DC Balanced SpaceWire**  
To be presented by Alex Kisin at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
What is a DC Balanced Communication Line?

- A DC balanced line is a communication line with “0” to “1” bits ratio over a certain time stretch equals to 1 (or very close to it)
  - DC component of DC balanced line is 0V or very close to it
  - This will allow data to pass through capacitive and transformer isolation barriers

- DC balanced lines allow using following types of SpaceWire interfaces:
  - Classical SpaceWire interface with Data and Strobe lines
    - When using it – both Data and Strobe has to be DC balanced
      - This creates extra efforts to comply with balancing 2 lines
  - Reduced set of interface cable created by eliminating Strobe lines
    - Only Data lines has to be DC balanced
    - There are numerous DC balancing schemes available with different levels of implementation complexity

New Approaches for DC Balanced SpaceWire
To be presented by Alex Kisin at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
Advantages of Using DC Balanced SpaceWire

- Greatly improved input common mode voltage (CMV) tolerance:
  - State-of-the-art Low Voltage Differential Signaling (LVDS) receivers tolerate only +5/-4V
    - Some systems may require much higher tolerances
  - CMV is limited only by capacitor or transformer voltage ratings (can be in quite high) when using these isolation techniques

- DC balanced lines allow using following types of SpaceWire interfaces:
  - Classical SpaceWire interface with Data and Strobe
    - When using it – both Data and Strobe must be DC balanced
  - Reduced set of interface cables created by eliminating Strobe lines
    - Only Data lines has to be DC balanced
      - No special tricks to balance 2 lines at a time
    - There are numerous DC balancing schemes available with different levels of implementation complexity
    - Cable and on-board hardware are reduced to 2 full-duplex differential pairs instead of 4
    - Smaller diameter cable – better bending radius
    - Less area required for interface components on PWB

New Approaches for DC Balanced SpaceWire
To be presented by Alex Kisin at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
Disadvantages of Using DC Balanced SpaceWire

- Greater data overhead for interfaces with both Data and Strobe
  - Balancing both Data and Strobe lines will require an increased amount of data traffic
    o Overhead amount greatly depends on used balancing methods
  - Some characters must be changed to keep DC balance for Data and Strobe lines
    o Null and Flag Control Character (FCT) characters may take new encoded values in character level
    o End of Packet (EOP), Error End of Packet (EEP) and Time codes may keep their original values because of their rare occurrences

- Easy clock extraction technique is gone for interfaces without Strobes
  - No Strobe systems will require to extract clock from an incoming encoded Data
    o More complex extracting techniques are required

- Easy on-a-fly data rate change is gone for interfaces without Strobes
  - System is supposed to know ahead of time at which rate it has to start
  - On-a-fly data rate change will require changes in an initial handshake process
    o This will require change of protocols

New Approaches for DC Balanced SpaceWire
To be presented by Alex Kis in at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
DC Balanced SpaceWire with Data and Strobe: Dual Bytes Method

- It is suggested to duplicate original Data byte with its inversed image
  - 2 bytes will be sent instead of 1
  - Each byte will have its Data Control Flag (DCF) and Parity bit which covers DCF and byte itself
    - 2-nd inversed byte will have inversed DCF and inversed Parity to maintain perfect balance

---

**Original Byte**

\[
\begin{array}{cccccccccc}
A & B & C & D & E & F & G & H \\
\end{array}
\]

- Split Original Byte into 2 bytes
- 1-st byte
- Inverted 2-nd byte with DCF+P

---

New Approaches for DC Balanced SpaceWire
To be presented by Alex Kisin at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
DC Balanced SpaceWire with Data and Strobe: Dual Bytes Method

Example

1 0 1 1 0 0 0 0

1-st byte

2-nd byte

Split Original Byte into 2 bytes

 DATA
 C LOCK
 STROBE

DCF

0 1 0 1 1 0 0 0 0 0 1 0 1 0 1 0 1 0 1 1

0 1 0 1 1 0 0 0 0 1 1 0 1 0 1 0 1 0 1 1

P

0 0 0 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1

0 0 0 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1

P

But for 20 bits...

 DATA: 1 0 '1' + 1 0 '0'
 STROBE: 7 '1' + 3 '0'

 DATA: 1 0 '1' + 1 0 '0'
 STROBE: 7 '1' + 3 '0'

New Approaches for DC Balanced SpaceWire
To be presented by Alex Kisin at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.

It is obvious that Data will be 100% DC balanced, but how about Strobe?

- Strobe is balanced too:
  - Because on a stretch of 20 bits clock will always place its “0” or “1” under complementary bits
    - Their resulted Exclusive OR (XOR) products will be inversion of each other and also be complementary
    - See colored columns above
Data and Strobe are balanced, but what to do with Control Characters?

- Control characters, except those used in handshake initialization, can either be:
  - Also inversely duplicated as described above (recommended), or
  - Used “as is”: they are so rare that small misbalance created by them will not be noticeable

- Control characters used in handshake initialization shall be both DC balanced for Data and Strobe:
  - Null can be send by an Originator for a long time when Responder is not ready
    - This might create significant DC bias on Strobe line because its Null image is misbalanced
  - Assign Null and FCT their new aliases when they are used in handshake protocol:
    - Substitute original Null character 01110100 with 10011100
      - Its resulted Strobe image will be changed from 00100001 to a DC balanced 11001001
    - Substitute original FCT character 0100 with 1100
      - Its resulted Strobe image will be changed from 0001 to a DC balanced 1001

New Approaches for DC Balanced SpaceWire
To be presented by Alex Kisin at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
What are Dual Bytes Method advantages?

- Very good DC balancing for Data and Strobe
- Easy implementation in FPGA
- “Single error correction” for an incoming data:
  - User can choose between 2 received bytes the one with valid parity
    - Only single bit errors can be detected, more errors may be masked
    - XOR both bytes to verify that there are no more than 1 bit error

What are Dual Bytes Method disadvantages?

- Doubled data rate
  - Will limit SpaceWire theoretical highest rate to half of what is possible
- Using aliases for control characters to mitigate their misbalancing
  - Small handshake protocol changes are required
What is PRS Method?

- The PRS method is intended to produce DC balanced data as a result of mixing random or pseudo-random bit sequence with real data stream
  - Mixing is done by the XOR function of both data streams on bit-by-bit basis
- Because it is difficult to produce a true random sequence – its pseudo-random version is used
  - PRS are usually generated by Linear Feedback Shift Registers (LFSR) (see backup slides)
- Resulted data stream is considered to be pseudo-random too (see backup slides)
  - All further XOR operations with a resulted PRS data stream will also produce PRS data stream
**PRS Method Implementation**

- It is suggested to start handshaking sequence with a disabled PRS generator
  - It will be difficult for an ad-hoc Responder receiver to sync to PRS generated by an Originator
    - Responder will not know what is a current state of PRS
  - Use the same Null and FCT characters balanced aliases as in above described Dual Byte method
- PRS can be enabled after both Originator and Responder exchanged with their Nulls and FCTs
- However, there are some requirements:
  - Disabled PRS startup seed shall generate “0” at its 1-st bit to be mixed with data (see backup slides)
    - Because both Null and FCT start with “1” – first “0” will indicate start of data cargo
  - PRS will be enabled immediately after start of data cargo is detected
  - No changes or aliasing from now on are required for an original Control characters and Time codes
  - Both Originator and Responder will keep tracking of both RX and TX PRS streams
DC Balanced SpaceWire with Data and Strobe: Pseudo-Random Sequence (PRS) Method (Continued)

Handshake Cycle At Any Pre-Selected Communication Rate

Data always starts with “0” after FCT

New Approaches for DC Balanced SpaceWire
To be presented by Alex Kisin at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
What are PRS Method advantages?

- Good DC balancing for Data and Strobe
- No data rate overhead: exactly same rate as an original data
- No changes for DCF and Parity bits are required
- No substitutes and aliases for control characters, except during handshaking

What are PRS Method disadvantages?

- Longer stretches of data to be consider balanced
- Moderate to complex implementation in FPGA
- PRS patterns matched with a same or inverted data patterns are possible
  - Probability of this is extremely low
  - May require LVDS receivers with wider CMV tolerances for DC offset mitigation
- Handshake protocol changes are required
DC Balanced SpaceWire with Data Only:
Dual Nibbles 4b/6b Coding

- It is suggested to substitute two of Data byte nibbles with two 6-bit symbols:
  - Each symbol will contain only 3 “0” and 3 ”1”
    - Number of permutations (P) for 3 “1” bits in 6-bit symbol for 64 symbols group is:
      \[ P(3) = \frac{6!}{(3! \times (6-3)!)} = 20 \]
      - It means that for each of 16 nibble’s combinations there is an unique 6-bit DC balanced symbol
  - There is no need anymore for Data Control Flag and Parity bit for this DC balancing method:
    - 4 remained symbols can be used as unique aliases for the original Control Characters
    - Data integrity can be checked by calculating number of “1” of a received 6-bit symbol: always 3
  - Overhead redundancy for this method (compared with the original SpaceWire) is only 20%:
    - Higher than for 4b/5b or 8b/10b methods (where overhead is 0%), but is much easier to implement
    - Lower than most of other Data Only methods
  - Requires lookup table of only 20 x 6 = 120 bits for transmitter, and 64 x 6 = 384 bits for receiver
    - If symbol doesn’t match table – it is an error
  - However, Strobe line can't be used because it is not DC balanced

New Approaches for DC Balanced SpaceWire
To be presented by Alex Kisin at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
Suggested symbols assignment for each 16 nibble combinations and 4 control characters:

<table>
<thead>
<tr>
<th>Data+Cont</th>
<th>MSB</th>
<th>&lt;--6-bit Symbol--&gt;</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1 0 1 1 1 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0 0 1 0 1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0 0 1 1 0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0 0 0 1 1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0 1 1 0 0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0 1 1 0 1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0 1 1 1 0</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0 1 0 1 1</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1 0 1 0 0</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1 0 0 0 1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data+Cont</th>
<th>MSB</th>
<th>&lt;--6-bit Symbol--&gt;</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>1 0 0 1 0</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>1 0 0 1 1</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1 1 1 1 0</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>1 1 0 0 1</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>1 1 0 1 0</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>1</td>
<td>0 1 0 0 1</td>
<td></td>
</tr>
<tr>
<td>EEP</td>
<td>1</td>
<td>1 1 1 0 0</td>
<td></td>
</tr>
<tr>
<td>EOP</td>
<td>0</td>
<td>0 0 1 1 1</td>
<td></td>
</tr>
<tr>
<td>FCT</td>
<td>1</td>
<td>0 1 0 1 0</td>
<td></td>
</tr>
<tr>
<td>ESC[ape]</td>
<td>0</td>
<td>1 0 1 0 1</td>
<td></td>
</tr>
</tbody>
</table>
What are Data Only w/ Fixed rates advantages?

- Only Data line balancing is required
- Reduced amount of cable
  - More space and less weight
  - Cheaper cables
- Sharper cable bend radius
  - A big plus for small satellites
- Less electronic hardware on-board:
  - Less power needed

What are Data Only w/ Fixed rates disadvantages?

- More complex methods of clock synchronization (see paper for suggestions)
- Longer stretches of data to be consider balanced
- More complex FPGA implementations
- Only single frequency data rate
  - No on-a-fly data rates changes
What are Data Only w/ Variable rates advantages?

- Same advantages as for Fixed data rates
- Variable rates are possible:
  - Use double handshake initialization
  - After initial low rate handshake is established:
    - Originator and Responder “tell” each other about new desired rate
    - Responder breaks connection for no less then 6.4us
    - Both Responder and Originator adjust their clock generators
    - Responder starts new handshake with a new rate

What are Data Only w/ Variable rates disadvantages?

- Same disadvantages as for Fixed data rates
  - Except rate changes
DC Balanced SpaceWire with Data Only:
4b/6b, Dual Bytes and PRS w/ Variable Data Rates Methods (Continued)

Handshake Cycles At
Variable Communication Rates

Originator

PRS Data @ New rate

Handshake @ New Rate

Responder

PRS Data w/ New Rate Negotiations

Handshake @ 10Mbps

Break >6.4μS

Handshake @ 10Mbps

PRS Data w/ New Rate Negotiations

Handshake @ New Rate

PRS Data @ New rate

New Approaches for DC Balanced SpaceWire
To be presented by Alex Kisin at the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Lines</th>
<th>RX Clock</th>
<th>Data Rate</th>
<th>Overhead vs. SpW</th>
<th>DC Balance</th>
<th>DC Quality</th>
<th>FPGA Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Standard SpaceWire</td>
<td></td>
<td></td>
<td></td>
<td>0%</td>
<td>No</td>
<td>Terrible</td>
<td>Existing</td>
</tr>
<tr>
<td>2</td>
<td>Dual bytes (DB) encoding</td>
<td>Data and Strobe</td>
<td>XOR-ed from D and S</td>
<td>Variable: as in original SpaceWire</td>
<td>100%</td>
<td>Yes</td>
<td>Very good</td>
<td>Easy</td>
</tr>
<tr>
<td>3</td>
<td>8b/20b</td>
<td></td>
<td></td>
<td></td>
<td>100%</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate-Complex</td>
</tr>
<tr>
<td>4</td>
<td>16b/30b</td>
<td></td>
<td></td>
<td></td>
<td>50%</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate-Complex</td>
</tr>
<tr>
<td>5</td>
<td>2 lines PRS modulation</td>
<td></td>
<td></td>
<td></td>
<td>0%</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate</td>
</tr>
<tr>
<td>6</td>
<td>Fixed DB or PRS modulation</td>
<td></td>
<td>4-phase sampling or others</td>
<td>Fixed: rate change requires dual handshaking</td>
<td>Same as in above D&amp;S</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate</td>
</tr>
<tr>
<td>7</td>
<td>Variable DB or PRS modulation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8b/10b or dual 4b/5b</td>
<td></td>
<td></td>
<td></td>
<td>0%</td>
<td>Yes</td>
<td>Very good</td>
<td>Moderate-Complex</td>
</tr>
<tr>
<td>9</td>
<td>Manchester modulation</td>
<td></td>
<td></td>
<td></td>
<td>100%</td>
<td>Yes</td>
<td>Excellent</td>
<td>Easy-Moderate</td>
</tr>
<tr>
<td>10</td>
<td>Dual nibble 4b/6b</td>
<td></td>
<td></td>
<td></td>
<td>20%</td>
<td>Yes</td>
<td>Very good</td>
<td>Easy-Moderate</td>
</tr>
</tbody>
</table>
LFSR Polynomial Generators for PRS

Here is a table of maximum-cycle Linear Feedback Shift Register (LFSR) taps. The bit numbering starts from \( n \) with \( n \) being the input bit and 1 the output bit. Figure 1 shows an 8-stage maximum-cycle LFSR. LFSR-2 refers to two tap LFSRs, LFSR-4 to LFSRs. Blanks indicate no solution exists.

Table 1: Shift Registers with Cycle Size \( 2^n - 1 \)

<table>
<thead>
<tr>
<th>( n )</th>
<th>LFSR-2</th>
<th>LFSR-4</th>
<th>n</th>
<th>LFSR-2</th>
<th>LFSR-4</th>
<th>n</th>
<th>LFSR-2</th>
<th>LFSR-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2, 1</td>
<td>4</td>
<td>24</td>
<td>24, 23, 21, 20</td>
<td>46</td>
<td>46, 40, 39, 38</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3, 2</td>
<td>25, 22</td>
<td>25</td>
<td>25, 24, 23, 22</td>
<td>47</td>
<td>47, 42, 47, 43, 42</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4, 3</td>
<td>26</td>
<td>26</td>
<td>26, 25, 24, 20</td>
<td>48</td>
<td>48, 44, 41, 39</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5, 3, 2</td>
<td>27</td>
<td>27</td>
<td>27, 26, 25, 22</td>
<td>49</td>
<td>49, 40, 49, 45, 44, 43</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6, 5, 3, 2</td>
<td>28</td>
<td>28, 25</td>
<td>28, 27, 24, 22</td>
<td>50</td>
<td>50, 48, 47, 46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7, 6, 5, 4</td>
<td>29</td>
<td>29, 27</td>
<td>29, 28, 27, 25</td>
<td>51</td>
<td>51, 50, 48, 45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8, 6, 5, 4</td>
<td>30</td>
<td>30</td>
<td>30, 29, 26, 24</td>
<td>52</td>
<td>52, 49, 52, 51, 49, 46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9, 8, 6, 5</td>
<td>31</td>
<td>31, 28</td>
<td>31, 30, 29, 28</td>
<td>53</td>
<td>53, 52, 51, 47</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10, 9, 7, 6</td>
<td>32</td>
<td>32</td>
<td>32, 30, 26, 25</td>
<td>54</td>
<td>54, 51, 48, 46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>11, 10, 9, 7</td>
<td>33</td>
<td>33, 20</td>
<td>33, 32, 29, 27</td>
<td>55</td>
<td>55, 31, 55, 54, 53, 49</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>12, 11, 8, 6</td>
<td>34</td>
<td>34</td>
<td>34, 31, 30, 26</td>
<td>56</td>
<td>56, 54, 52, 49</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>13, 12, 10, 9</td>
<td>35</td>
<td>35, 33</td>
<td>35, 34, 28, 27</td>
<td>57</td>
<td>57, 50, 57, 55, 54, 52</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>14, 13, 11, 9</td>
<td>36</td>
<td>36, 25</td>
<td>36, 35, 29, 28</td>
<td>58</td>
<td>58, 39, 58, 57, 53, 52</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>15, 14, 13, 11, 9</td>
<td>37</td>
<td>37</td>
<td>37, 36, 33, 31</td>
<td>59</td>
<td>59, 57, 55, 52</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>16, 14, 13, 11</td>
<td>38</td>
<td>38</td>
<td>38, 37, 33, 32</td>
<td>60</td>
<td>60, 59, 60, 58, 56, 55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>17, 16, 15, 14</td>
<td>39</td>
<td>39, 35</td>
<td>39, 38, 35, 32</td>
<td>61</td>
<td>61, 60, 59, 56</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>18, 17, 16, 13</td>
<td>40</td>
<td>40</td>
<td>40, 37, 36, 35</td>
<td>62</td>
<td>62, 59, 57, 56</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>19, 18, 17, 14</td>
<td>41</td>
<td>41, 38</td>
<td>41, 40, 39, 38</td>
<td>63</td>
<td>63, 62, 63, 62, 59, 58</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>20, 19, 18, 16</td>
<td>42</td>
<td>42</td>
<td>42, 40, 37, 35</td>
<td>64</td>
<td>64, 63, 61, 60</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>21, 20, 19, 16</td>
<td>43</td>
<td>43</td>
<td>43, 42, 38, 37</td>
<td>65</td>
<td>65, 47, 65, 64, 62, 61</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>22, 21, 20, 17</td>
<td>44</td>
<td>44</td>
<td>44, 42, 39, 38</td>
<td>66</td>
<td>66, 60, 58, 57</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>23, 22, 20, 18</td>
<td>45</td>
<td>45</td>
<td>45, 44, 42, 41</td>
<td>67</td>
<td>67, 66, 65, 62</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
LFSR Hardware Implementation for PRS

"0" Seed startup value

Figure 1: An 8-stage Galois LFSR with cycle size 255. This LFSR has taps at positions 8, 6, 5 and 4.

Requirements to LFSR generator:

- Start seed shall have a "0" at Data XOR mixer (shown above) when LFSR is disabled
- All "0" seeds are prohibited
- All "1" seeds should be skipped to improve DC balance
  - Detect LFSR state prior to all "1" state
  - On a next clock cycle upload LFSR with state that follows after all "1"
    - This will skip unwanted all "1" state
Let $A$ be an array of $N$ bits containing the message and $R$ be a random bit array of the same length with individual bits being independent and taking values 0 and 1 with probability of $\frac{1}{2}$. Then a bitwise XOR operation (denoted by $\oplus$) applied to $A$ and $R$ will result in an array $X = A \oplus R = R \oplus A$ which is random and distributed the same way as $R$ (i.e. its individual bits are independent and take values 0 and 1 with a same probability of $\frac{1}{2}$).

**PROOF**

Consider arbitrary bit $X_k = A_k \oplus R_k = (A_k + R_k) \mod (2)$. Resulted conditional probabilities for various $A_k$ and $R_k$ are:

- $P\{X_k = 0 | A_k = 0\} = P\{R_k = 0\} = \frac{1}{2}$
- $P\{X_k = 0 | A_k = 1\} = P\{R_k = 1\} = \frac{1}{2}$
- $P\{X_k = 1 | A_k = 0\} = P\{R_k = 1\} = \frac{1}{2}$
- $P\{X_k = 1 | A_k = 1\} = P\{R_k = 1\} = \frac{1}{2}$

Therefore, resulted probabilities of $P\{X_k = 0\} = P\{X_k = 1\} = \frac{1}{2}$.

Consider now arbitrary but different 2 bits $X_m$ and $X_n$. We have:

- $P\{X_m = 0, X_n = 0 | A_m = 0, A_n = 0\} = P\{R_m = 0, R_n = 0\} = P\{R_m = 0\}* P\{ R_n = 0\} = \frac{1}{4}$ due to the independence of $R_m$ and $R_n$.

Repeating this calculation 15 more times for all possible combinations of values of $X_m$, $X_n$, $A_m$, and $A_n$, we can find that all the mutual probabilities $P\{X_m = U, X_n = V\}$ with $U, V = 0$ or 1 are equal to $\frac{1}{4}$. So, for any combination of $U$ and $V$ we have:

- $P\{X_m = U, X_n = V\} = \frac{1}{4} = \frac{1}{2} * \frac{1}{2} = P\{X_m = U\} * P\{X_n = V\}$.

**CONCLUSIONS**

Therefore, $X_m$ and $X_n$ are independent and a whole resulted bit array $X$ is random. All further XOR mixing of random bit array $X$ with bit arrays similar to $A$ will also produce random bit arrays $X'$.

In practice pseudorandom array PR does not strictly satisfies these conditions, so the result is valid to the extent of PR being close to random vector of independent bits taking values 0 and 1 with probability $\frac{1}{2}$. The longer generated PR sequence – the closer it approaches to $R$. LFSR registers can be good example of PR generators.

**HISTORY**

The above concept of mixing together general data with random data and still get random data was first suggested by Frank Miller in 1882 and “re-invented” and patented in 1917 by Gilbert S. Vernam as “a secure interceptible spy communications method, aka “One Time Pad”. Refer to: [https://en.wikipedia.org/wiki/One-time_pad](https://en.wikipedia.org/wiki/One-time_pad)