TSG-RAN WG-1
Meeting # 7
Hannover, Germany
Aug.30-Sept.3, 1999

TSG-R1#7(99)Cxx

Title: Clarifications on Golay-Hadamard Sequence Based RACH Preamble

Source: Nortel Networks[1]

0.0 Summary

In this contribution, further clarifications are made for Golay-Hadamard preamble proposed in TSGR1#6(99)990. The key advantages of the proposed Golay-Hadamard preamble are:

- It allows very efficient implementation of RACH detector at Node B. **Complexity reduction** can be achieved by employing so-called Efficient Golay Correlator as compared to the straight-forward PN correlator;
- It allows a very **flexible preamble generation** at very low cost;
- It also allows a **fully compatible with** parallel correlator implementation for the PN-Hadamard preamble without hardware change;
- It is fully compatible with the current PAR reduction preamble modulation scheme;
- It has lower auto-correlation side-lobes than the PN-Hadamard preamble;
- It retains the same Doppler and Fading resistant capability as the PN-Hadamard preamble.

1 Contact Person: Catherine Gauthier, Wen Tong, Nortel Networks
1 Place des Freres Montgolfier, 78042 Guyancourt, France
Tel:+33 1 39 44 57 47
Fax:+33 1 39 44 50 12
e_mail: gauth@nortel.com

0.1 Golay-Hadamard Preamble Definition

0.1.1 Structure of the Golay-Hadamard Preamble

In TSGR1#6(99)990, a method to avoid the excessive high energy of the inter-signature cross-correlation of Golay sequence based preamble (TSGR1#3(99)205) was proposed. By introducing 4 interleaving transformations, a pair of constituent Golay sequence of 256 chips can be transformed into other 7 pairs of Golay sequences each with an ideal aperiodic auto-correlation. These interleaved Golay sequence are concatenated and extended into 4096 chips long scrambling code, without repeating the constituent Golay sequence. The 4 types of interleaves: T1, T2, T3, T4 are defined in TSGR1#3(99)990. The procedure to generate the RACH scrambling code is as follows:

- Step-1: Generate constituent pair A,B (as in TSGR1#3(99)205)
- Step-2: Apply interleave transforms to A,B, generating the extended Golay sequence **G**:
  A,B,T1(A),T1(B),T2(A),T2(B),T1(T2A)),T1(T2(B)),T3(A),T3(B),
  T1(T3(A)),T1(T3(B)),T4(A),T4(B),T1(T4(A)),T1(T4(B))
- Step-3: Select a 16 chip Hadamard sequence, concatenate such a sequence 256 times to generate the signature sequence.
- Step-4: Use signature in Step-3 to cover the scrambling code in Step-2 to obtain the preamble.

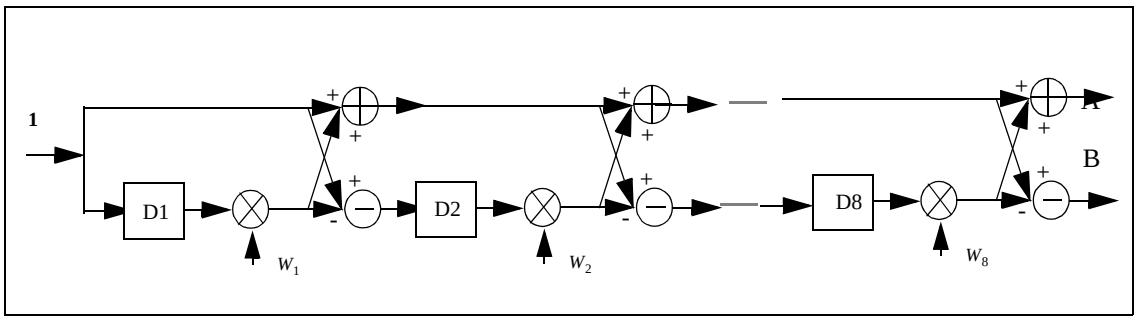
Therefore, the entire 4056-chip long scrambling code is build on a 256-chip pair of constituent pair A,B which is defined by vector P and W (see TSGR1#3(99)205). The P vector is optimized such that the cross correlation between the pairs are minimized and the W vector is the cell-specific vector of 8-bit long.

0.2 Hardware Complexity Evaluations

0.2.1 Golay-Hadamard Preamble Generation at UE

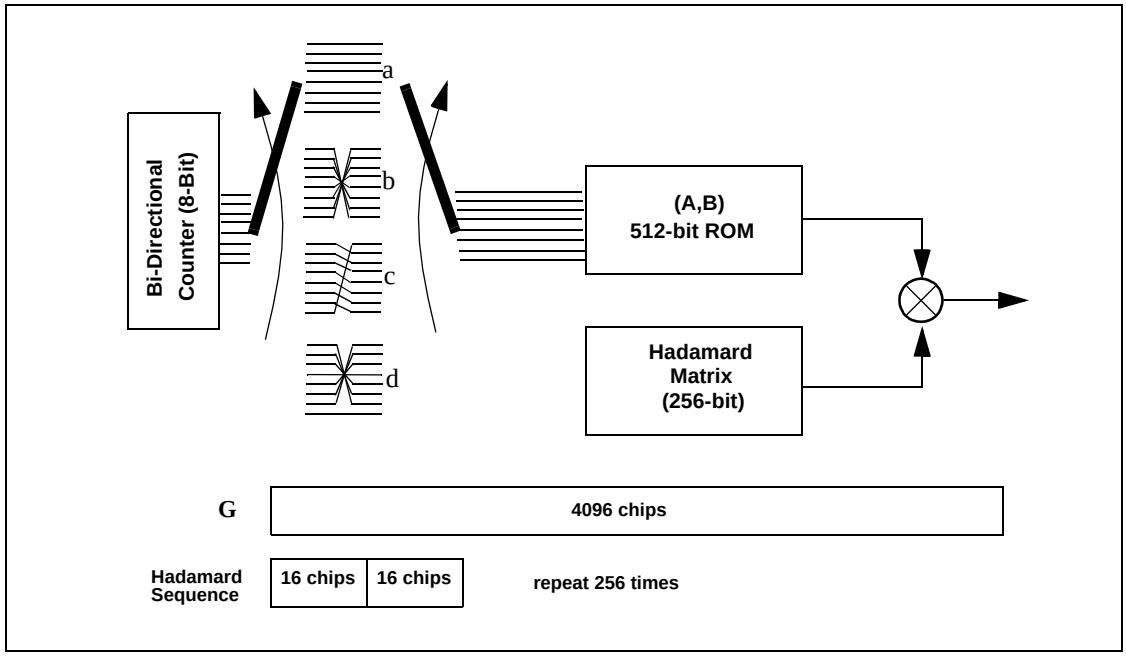
The constituent Golay pair A,B can be generated by using well known binary lattice structure which contains 8-section of lattice shown in Figure 1. The input is Kronecker delta function.

FIGURE 1. Fast Recursive Golay Sequence Generator



By re-configuring the W parameter, such a generator can generate a set of 256 cell specific constituent Golay pairs A,B. Another design alternative is to use soft ware to generate A,B. and use interleaving operation to generate 4096-chip long preamble as shown in Figure 2. The introduced for interleave transform can be implemented at a very low cost by simply re-configuring the address bus connections.

FIGURE 2. UE Transmitter



The detailed generation procedure (time sequence) is listed in Table 1. The preamble generation is simple control the cyclic switch connection to bus a,b,c,d, and counter up/down to read the 512 bit pre-stored pair (A,B) .

TABLE 1. Golay-Hadamard Scrambling Code

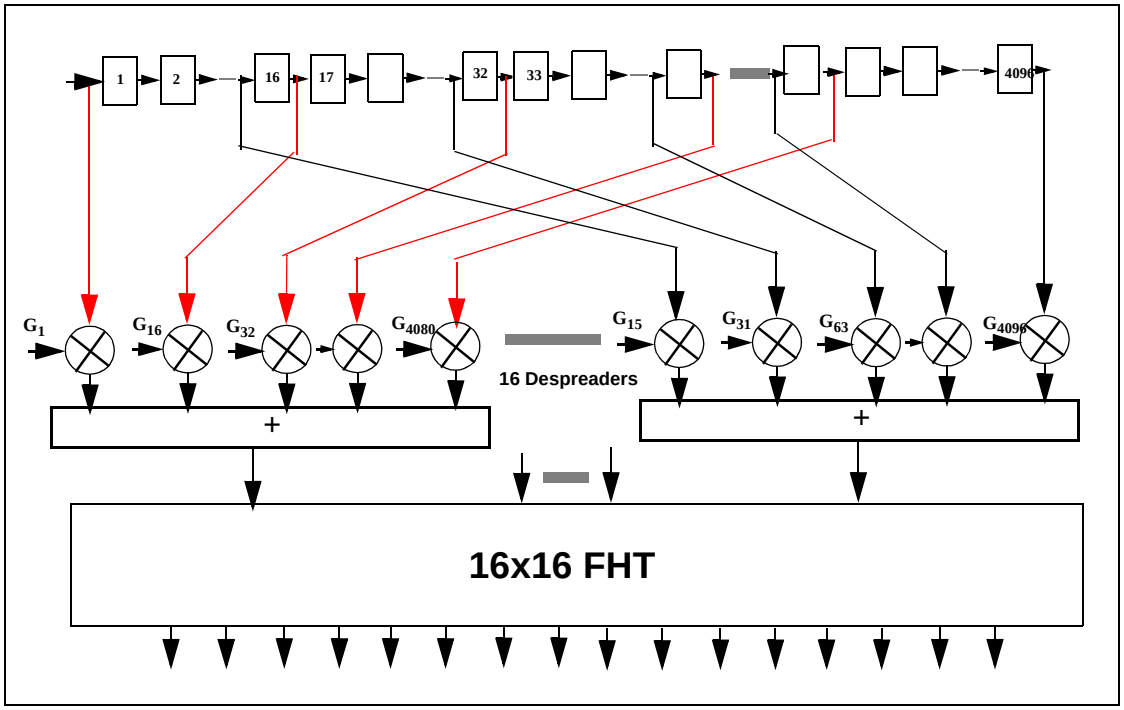
chips (G)	1-512	513-1024	1025-1536	1537-2048	2049-2560	2561-3072	3073-3584	3585-4096
sequence	A,B	T1(A),T1(B)	T2(A),T2(B)	T1(T2(A)),T1(T2(B))	T3(A),T3(B)	T1(T3(A)),T1(T3(B))	T4(A),T4(B)	T1(T4(A)),T1(T4(B))
counter	up	down	up	down	up	down	up	down
bus switch	a		b		c		d	

The advantage of the of Golay-Hadamard preamble is that it only requires 8-bit to represent cell-specific scrambling codes, i.e. the UE can use 8-bit cell-specific scrambling code received from BCH to generate entire preamble, while for the PN-Hadamard preamble, we have to either down load the 2x24 initial setting values for the PN generator from BCH by adding more signalling overheads, or to store the 2x24x256 bits of initial PN generator setting values, in order to generate all 256 cell-specific scrambling codes. By taking into account of this factor, even we will not re-use the UL scrambling code generator, the proposed Golay-Hadamard preamble generator adds no additional complexity to UE.

0.2.2 Golay-Hadamard Preamble Detector at BTS

The PN-Hadamard preamble detector at Node B proposed in TSGR1#6(99)983 is shown Figure 3. This is based on the Walsh-Hadamard signature repetition and covering with PN scrambling code in order to have good Doppler resistant capability. As we can see, each Hadamard chip has a full range of time diversity of PN chips on RACH detector side. It can be seen from Figure 3, for each de-spreader the distance of the received PN sequence is 16, therefore a direct reuse of the uplink de-spreader is not possible. Note that **by replacing the correlator coefficients of PN sequence C by extended Golay sequence G, the preamble detector in Figure 3 become the Golay-Hardamard preamble detector.**

FIGURE 3. Base-Station RACH Preamble Detector Architecture



On the other hand, the cell-specific PN scrambling code used as re-configureable correlator coefficients can not be generated on-the-fly, since it requires the PN generator operates at 256 times chip rate Fc. Therefore the entire 4096 PN chips must pre-generated and stored.

For Golay-Hadamard preamble, we can generated the correlator coefficients on-line. This can be done by running the address bus switching at chip rate Fc and the counter at clock rate of Fc/16. (Figure 2). Table 2 lists the complexity of correlator based RACH preamble detector shown in Figure 3. As we can see the major complexity part the correlator with 8 level of adder trees and sign multipliers. However, by exploiting to the Golay sequence structure, a so-called Efficient Golay Correlator (EGC) can be employed (TSGR1#3(99)205), a 225-tap correlator requires 225 sign multipliers while EGC requires

TABLE 2. Generic RACH Preamble Match Filter Complexity

Units	Parallel Architecture (Fig.5)
Shift Registers	16x256=4096
Sign Multiplier	16x256=4096
Adders	16x255=4080
Adders (FHT)	16x4 (16-Signatures)

only 8, correlator requires 254 address while EGC requires only $2\log_2(256)=16$. *EGC can result in about 16 times complexity reduction.*

0.3 Performance Evaluation

The proposed Golay-Hadamard preamble is compared with PN-Hadamard preamble proposed in TSGR1#6(99)983. In what follows, we present the a preliminary simulation results of the Golay-Hadamard preamble and its comparison with PN-Hadamard preamble.

0.3.1 Correlation Properties of the Golay Hadamard Preamble

One of the major advantage of the Golay complementary sequence is its excellent auto-correlation. Since the proposed Golay preamble scrambling code is a concatenated of 16 short length Golay pair sequences, the extended Golay preamble scrambling code has possesses a lower auto correlation side-lobe that PN code preamble scrambling code. Figure 4 shows the distribution of maximum absolute value of side-lobe among 16 signatures within w.r.t. a cell-specific scrambling code. It can be seen that under different search window the Golay preamble scrambling sequence has better auto a-periodic correlation.

FIGURE 4. Comparison of aperiodic auto-correlation of Golay-WH and PN-WH preamble

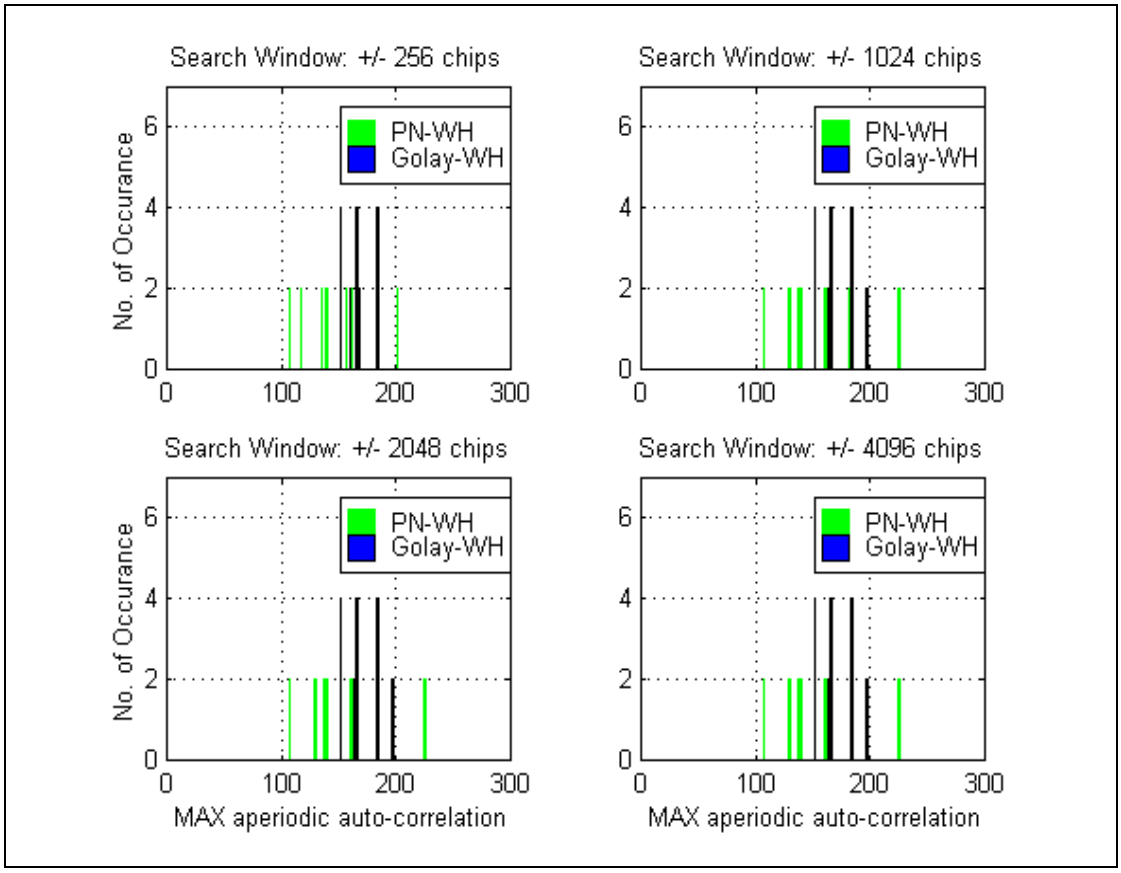
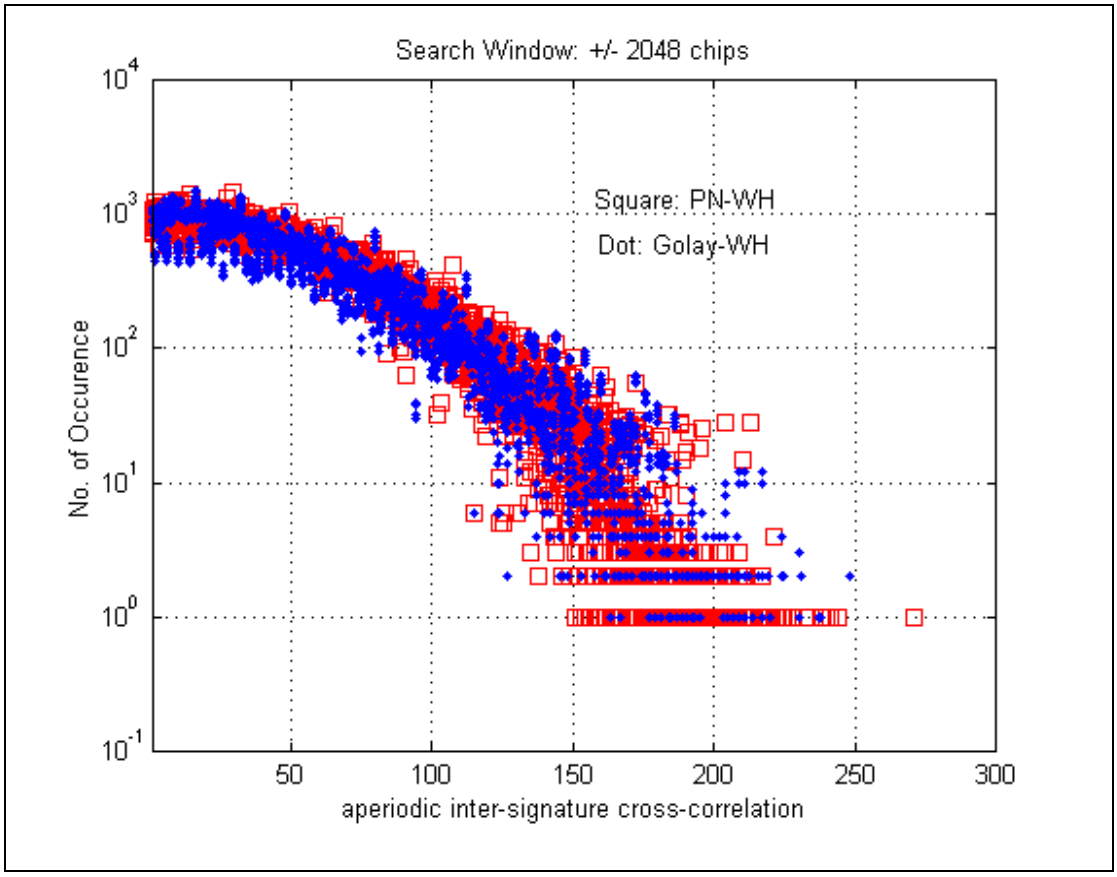


Figure 5 shows the inter-signature cross-correlation distribution for both Golay and PN preamble scrambling code. As we can see, both Golay and PN preamble scrambling code have similar cross-correlation performance.

FIGURE 5. Inter-Signature Cross-Correlation for Golay-WH and PN-WH preambles



0.3.2 High Speed Fading and High Speed Doppler Detection Performance Simulations

Since the proposed the Golay-Hadamard preamble has the same signature structure as the Motorola/TI PN-Hadamard structure (TSGR1#7(99)983). Simulations have been performed to demonstrate that the proposed Golay-Hadamard preamble has the same Doppler resistant performance as PN-Hadamard preamble.

The simulations are conducted under the following conditions, an arbitrary select preamble is transmitted with a transmission delay uniformly chosen within a window of 956 chips (cell size of 35 km), the decision statistics of the transmitted preamble is computed by using coherent accumulation for each Hadamard code (signature) and each position of the searching window. The maximum of the these statistics is selected and then compared to a threshold to make a hypothesis test to evaluate the preamble signature detection probability. The threshold is determined such that the false alarm probability is set to 0.001.

Two types tests are performed: (1) High frequency offset (2) High speed Rayleigh fading. In Figure 6,7 the preamble detection in the presence of high frequency offset are presented for both Golay-Hadamard signature and PN-Hadamard signature. In Figure 8,9 the high speed Rayleigh fading channel detection performance are presented.

FIGURE 6. Preamble detection in the presence of frequency offset

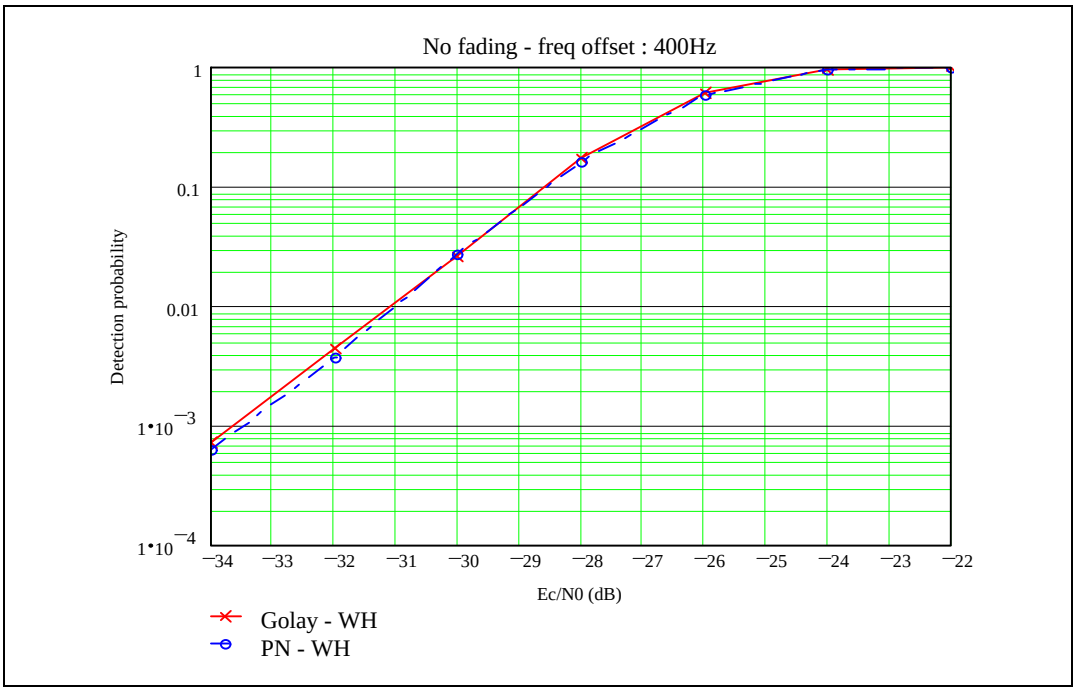


FIGURE 7. .Preamble detection in the presence of frequency offset

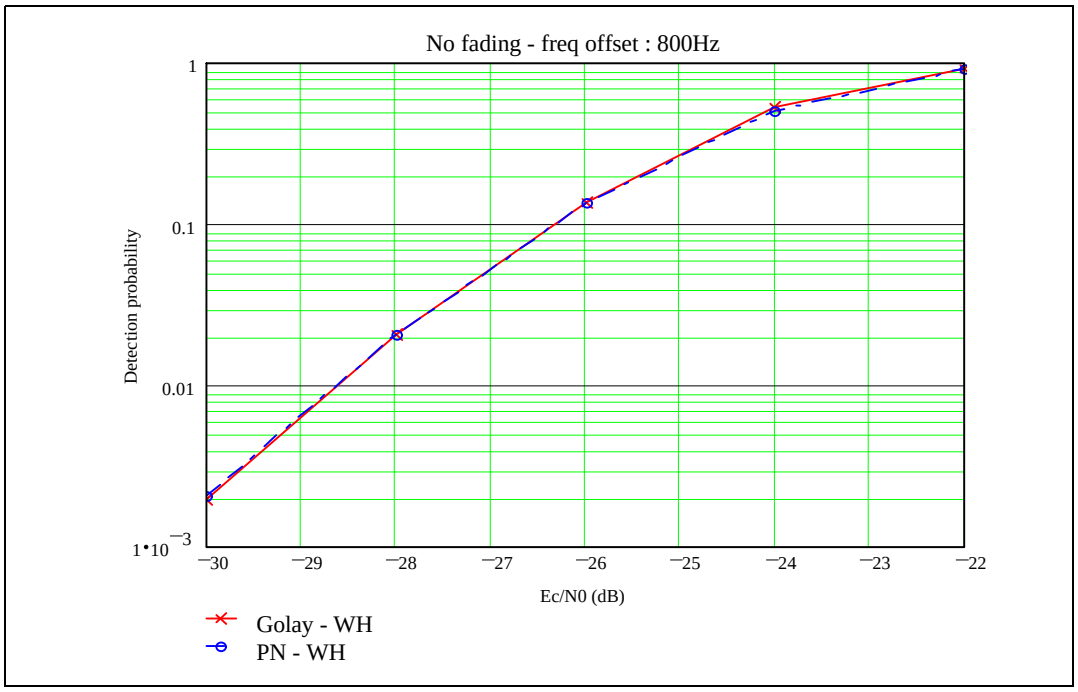


FIGURE 8. Preamble detection in the presence of high speed Rayleigh fading

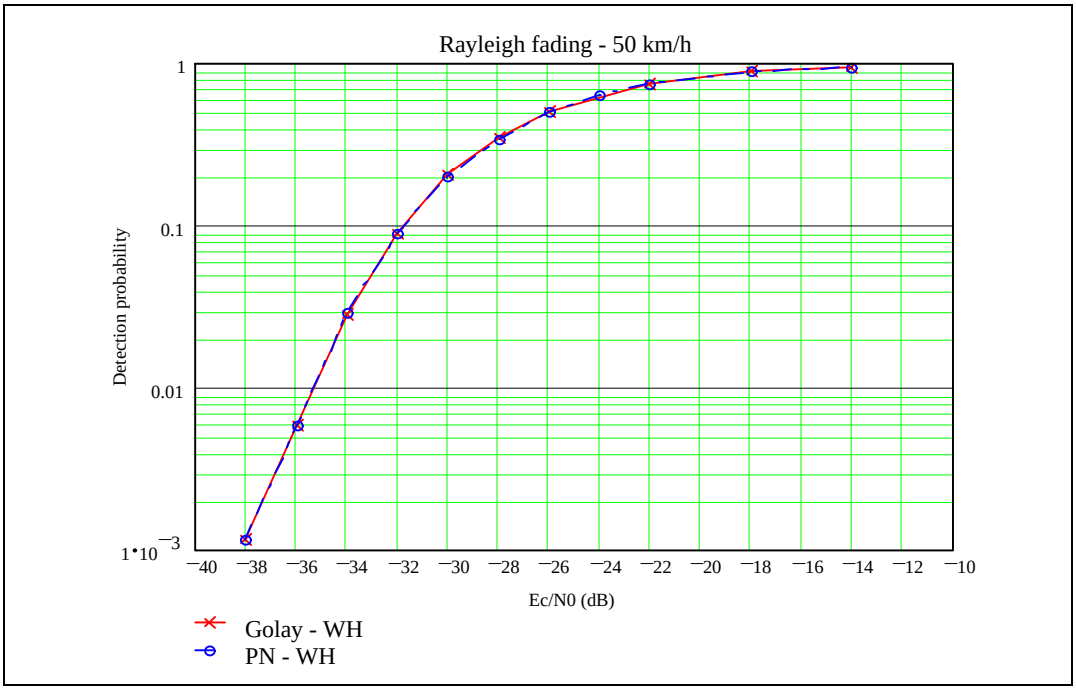
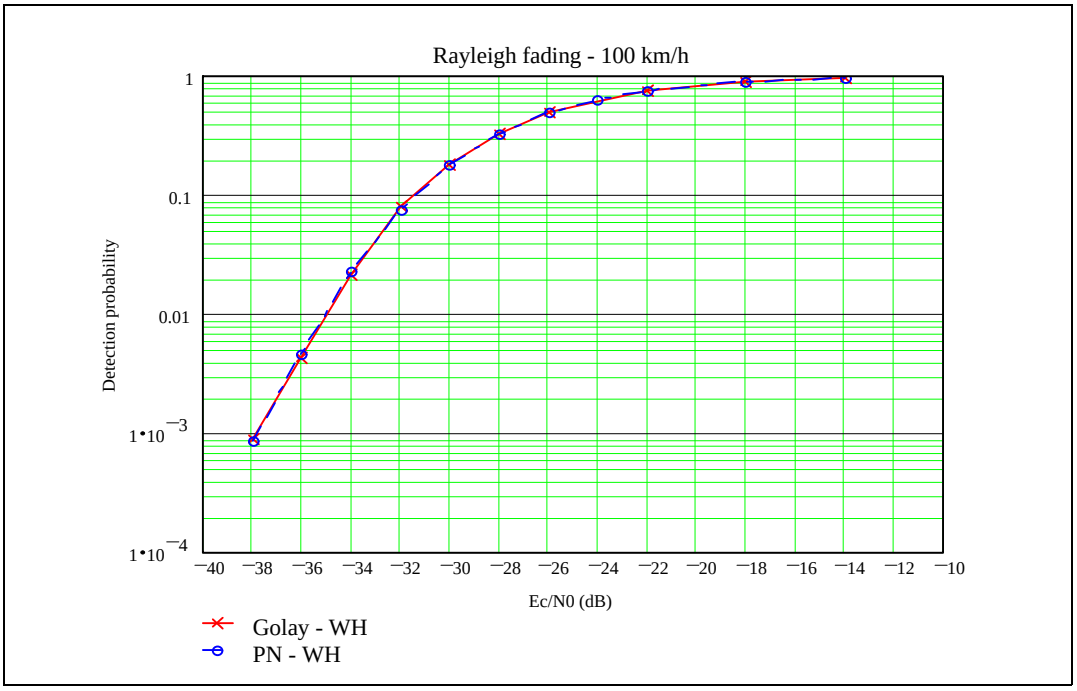


FIGURE 9. Preamble detection in the presence of high speed Rayleigh fading



It can be seen that the Golay Hadamard preamble possesses the same Doppler and fading resistant property as the PN-Hadamard preamble.

0.4 Summary and Conclusions

In this contribution, further clarifications on the Golay-Hadamard preamble are made. The key advantage of the proposed Golay-Hadamard preamble are:

- It allows very efficient implementation of RACH detector at Node B. A significant **complexity reduction** can be achieved by employing so-called Efficient Golay Correaltor as compared to the straight forward PN correlator.
- It allows a very **flexible preamble generation** at very low cost.
- It also allows and is **fully compatible with** parallel correlator implementation for the PN-Hadamard preamble without hardware change.
- It is fully compatible with the current PAR reduction preamble modulation scheme
- It has lower auto-correlation side-lobes than the PN-Hadamard preamble
- It retains the same Doppler and Fading resistant capability as the PN-Hadamard preamble.

We recommend to replace the PN preamble scrambling code by the extended Golay sequence proposed in TSGR1#6(99)990.

Appendix A Text Proposal

- Step-1: Generate constituent pair $A = (a_0, a_1 \ldots a_{255})$, $B = (b_0, b_1 \ldots b_{255})$ using following recursive relation:

$$a_0(k) = \delta(k)$$
$$b_0(k) = \delta(k)$$
$$a_n(k) = a_{n-1}(k) + W_n(v, n) b_{n-1}(k - 2^{P_n})$$
$$b_n(k) = a_{n-1}(k) - W_n(v, n) b_{n-1}(k - 2^{P_n})$$

  where, $\delta(k)$ is the Kronecker delta function, $k = 0, 1, \ldots, 255$

  $n = 1, 2, \ldots 8$, $P_n = (0, 2, 5, 7, 6, 1, 4, 3)$ and $W_n(v, n) = (-1)^{B_n(v)}$, with $v = 0, 1, \ldots 255$ and $B_n(v)$ is the n-th bit of 8-bit long binary representation of integer v.

- Step-2: Apply interleave transforms T1,T2,T3,T4 to A,B, generating the extended Golay sequence $\boldsymbol{G}$:
  A,B,T1(A),T1(B),T2(A),T2(B),T1(T2A)),T1(T2(B)),T3(A),T3(B), T1(T3(A)),T1(T3(B)),T4(A),T4(B),T1(T4(A)),T1(T4(B))
  where, $T1(A) = (a_{255}, a_{254}, \ldots a_1, a_0)$

$$T2(A) = (a_{BR(0)}, a_{BR(1)} \ldots a_{BR(255)})$$
$$T3(A) = (a_0, a_2 \ldots a_{254}, a_1, a_3 \ldots a_{255})$$
$$T4(A) = (T2(A_1) T2(A_2))$$

  and $A_1 = (a_0, a_1 \ldots a_{127})$, $A_2 = (a_{128}, a_{129} \ldots a_{255})$

  $BR(v)$ denotes reverse the bit ordering of binary representation of integer v.

- Step-3: Select a 16 chip Walsh-Hadamard sequence $h^{(k)}$ as k-th signature, concatenate such a sequence 256 times to generate the signature sequence $\boldsymbol{H} = (h_0^{(k)} h_1^{(k)} \ldots h_{255}^{(k)})$.

- Step-5: Use signature in Step-3 to cover the scrambling code in Step-2 to obtain the preamble. $\boldsymbol{G} \otimes \boldsymbol{H}$,