Πανεπιστημιο Πατρων

Μηχανικων Ηλεκτρονικων Υπολογιστων & Πληροφορικης

Ψηφιακές Τηλεπικοινωνίες: 2η Εργαστηριακή Άσκηση

Σύγκριση Ομόδυνων Ζωνοπερατών Συστημάτων 8-PSK και 8-FSK

Συγγραφείς:
Κων/νος Αραβανής ΑΜ: 3628
Δημήτρης Λεβεντέας ΑΜ: 3688

Επιβλέπων:
Κων/νος Μπερμπερίδης

9 Μαρτίου 2009

Περιεχόμενα

1 Εισαγωγή

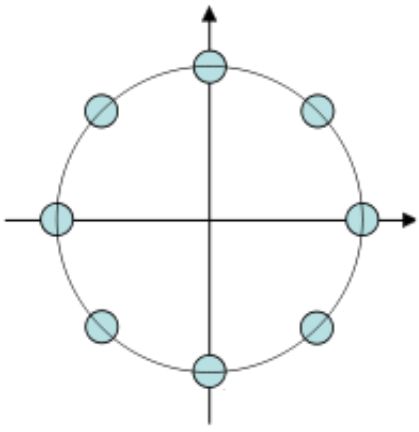
Στην παρούσα εργασία μελετάμε την διαβίβαση της ψηφιακής πληροφορίας μέσα από κανάλια επικοινωνίας, τα οποία χαρακτηρίζονται ως κανάλια προσθετικού λευκού Gaussian θορύβου (Additive White Gaussian Noise - AWGN). Τα κανάλια αυτά είναι βασικά αναλογικά, με άλλα λόγια η προς μετάδοση ψηφιακή πληροφορία πρέπει να απεικονιστεί σε αναλογικές κυματομορφές σήματος πληροφορίας. Για την μελέτη αυτή, προχωρήσαμε στην σύγκριση των διαμορφώσεων 8-PSK και 8-FSK ως προς την απόδοση τους. Οι συγκρίσεις αυτές βασίστηκαν σε μετρήσεις πιθανότητας σφάλματος bit (Bit Rate Error (BER) και συμβόλου Symbol Error Rate (SER) που πραγματοποιήθηκαν σε ομόδυνα ζωνοπερατά συστήματα με την χρήση ορθογώνιου παλμού.

2 Βασικά χαρακτηριστικά PSK

Εάν έχουμε ένα σύνολο M δισδιάστατων κυματομορφών σήματος, έστω $s_m(t), m = 1, 2, 3, ..., M$ μπορούμε να δημιουργήσουμε ένα σύνολο M ζωνοπερατών κυματομορφών ως:

$$u_m(t) = s_m(t) \cos 2\pi f_c t \tag{1}$$

Στην ειδική περίπτωση M δισδιάστατων ζωνοπερατών κυματομορφών με την ίδια ενέργεια, τα αντίστοιχα σημεία σήματος στη γεωμετρική αναπαράσταση των κυματομορφών ανήκουν σε ένα κύκλο ακτίνας $\sqrt{E_s}$ όπως φαίνεται στο σχήμα 1.



Σχήμα 1: Αστερισμός σήματος $M = 8$ σημείων που αντιστοιχεί σε σύνολο διορθογώνιων κυματομορφών με ίδια ενέργεια

Από την γεωμετρική αναπαράσταση για $M = 8$, παρατηρούμε ότι τα σημεία σήματος είναι ισοδύναμε με ένα μοναδικό σήμα, του οποίου η φάση ολισθαίνει

κατά πολλαπλάσια του $\frac{\pi}{4}$. Με άλλα λόγια, ένα ζωνοπερατό σήμα της μορφής

$$s(t)\cos 2\pi f_c t + \frac{\pi m}{4}, \qquad m = 1, 2, ..., 8 \tag{2}$$

έχει την ίδια γεωμετρική αναπαράσταση με ένα σύνολο $M = 8$ διορθογώνιων σημάτων. Επομένως ένας απλός τρόπος για την δημιουργία M ζωνοπερατών σημάτων με την ίδια ενέργεια είναι να αποτυπώσουμε την πληροφορία στην φάση του φέροντος. Έτσι, έχουμε ένα διαμορφωμένο κατά την φάση-φέροντος σήμα. Η γενική αναπαράσταση ενός συνόλου M διαμορφωμένων κατά φάση-φέροντος κυματομορφών είναι:

$$u_m(t) = g_T(t)\cos(2\pi f_c t) + \frac{2\pi m}{M}, \qquad m = 0, 1, ..., M-1 \quad \text{και} \quad 0 \leq t \leq T \tag{3}$$

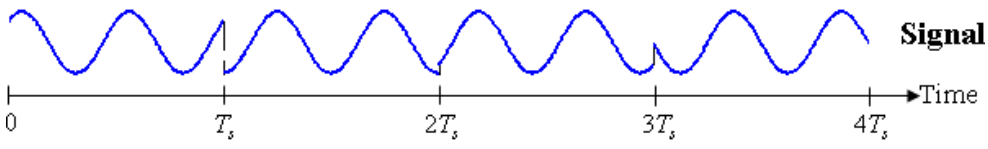
όπου $g_T(t)$ είναι ένας παλμός βασικής ζώνης για την μορφοποίηση ο οποίος καθορίζει τα φασματικά χαρακτηριστικά του μεταδιδόμενου σήματος. Όταν ο $g_T(t)$ είναι ένας ορθογώνιος παλμός, ο οποίος ορίζεται ως

$$g_t(T) = \sqrt{\frac{2E_s}{T}}, \qquad 0 \leq t \leq T \tag{4}$$

Οι αντίστοιχες μεταδιδόμενες κυματομορφές σήματος γίνονται:

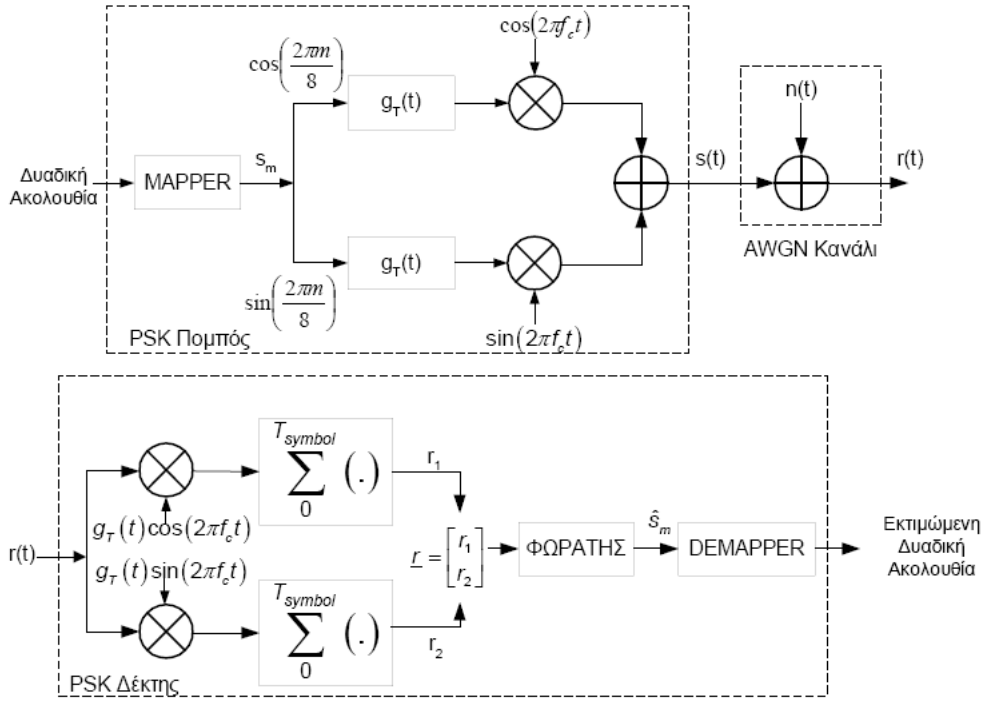
$$u_m(t) = \sqrt{\frac{2E_s}{T}}\cos(2\pi f_c t) + \frac{2\pi m}{M}, \qquad m = 0, 1, ..., M-1 \quad \text{και} \quad 0 \leq t \leq T \tag{5}$$

Έχουν σταθερή περιβάλλουσα και η φάση του φέροντος αλλάζει απότομα στην αρχή κάθε διαστήματος σήματος. Αυτός ο τύπος ψηφιακής διαμόρφωσης καλείται μεταλλαγή ολίσθησης φάσης Phase Shift Keying (PSK).



Σχήμα 2: Παράδειγμα ενός τετραδικού PSK σήματος με $T = \frac{2}{f_c}$

Το σχήμα 2 δείχνει ένα PSK τεσσάρων φάσεων το οποίο συνήθως καλείται ορθογώνιο PSK (Quadrature Phase Shift Keying (QPSK) σήμα.



Σχήμα 3: Αναπαράσταση ομώδυνου 8-δικού PSK

2.1 Κύκλωμα PSK

Στο σχήμα 3 απεικονίζεται ο σχεδιασμός ενός PSK κυκλώματος. Συγκεκριμένα μπορούμε να διακρίνουμε τα διάφορα modules που το απαρτίζουν.

Αρχικά, παράγεται η δυαδική ακολουθία η οποία διοχετεύεται στον mapper όπου και γίνεται η αντιστοίχηση της σε σύμβολα. Ο mapper έχει την δυνατότητα να κάνει κωδικοποίηση κατά Gray ώστε να επιτύχουμε γειτονικά στο δισδιάστατο χώρο σημάτων σύμβολα, να αντιστοιχούν σε ακολουθίες bit που διαφέρουν κατά μικρό αριθμό bits.

Η έξοδος του mapper διοχετεύεται με την σειρά της στον modulator ο οποίος διαμορφώνει την κάθε συνιστώσα . Με άλλα λόγια τις πολλαπλασιάζει με τον ορθογώνιο παλμό και την διαμορφώνει γύρω από τη φέρουσα συχνότητα ώστε να προκύψει το επιθυμητό ζωνοπερατό σήμα που περιγράφεται από την συνάρτηση (5).

Στο σχήμα επίσης παρουσιάζεται και το κανάλι AWGN που πρόκειται για τον θόρυβο που προκύπτει κατά την διάρκεια μετάδοσης του σήματος από τον πομπό του δέκτη. Στα πειράματα μας, ο θόρυβος είναι μηδενικής μέσης τιμής και διασποράς $\sigma = \frac{N_0}{2}$.

Με την σειρά του ο δέκτης, το λαμβανόμενο σήμα το αποδιαμορφώνει με τον demodulator. Για να το κάνει αυτό, θα πρέπει ο δέκτης να γνωρίζει την

φάση της φέρουσας και τα χρονικά πλαίσια κάθε συμβόλου, δηλαδή να είναι πλήρως συγχρονισμένος με τον πομπό. Με άλλα, λόγια κατά την προσομοίωση θεωρούμε ότι το 8-PSK είναι ομόδυνο. Ο αποδιαμορφωτής συσχετίζει το ληφθέν σήμα με τις δύο συνιστώσες της φέρουσας, οπότε προκύπτουν δυο τιμές, δηλαδή ένα διάνυσμα r το οποίο αποτελεί την εκτιμηθείσα τιμή του τρέχοντος συμβόλου πάνω στον αστερισμό του 8-PSK. Η συσχέτιση γίνεται στα χρονικά πλαίσια μιας περιόδου συμβόλου.

Στην συνέχεια ο φωρατής δέχεται ως είσοδο το διάνυσμα r και αποφασίζει σε ποιο σύμβολο βρίσκεται εγγύτερα. Το διάνυσμα s_m που θα έχει την μικρότερη απόσταση από το r αντιστοιχεί και στο σύμβολο που στάλθηκε (φωρατής μέγιστης πιθανοφάνειας).

Τέλος, παίρνοντας στον demapper γίνεται η αντιστοίχηση στην εκτιμώμενη απεσταλθήσα δυαδική ακολουθία.

Ο κώδικας με τον οποίο εξομοιώσαμε τον παραπάνω τύπο διαμόρφωσης, παρατίθεται στο παράρτημα.

3 Βασικά χαρακτηριστικά FSK

Το 8-δικό FSK μπορεί να χρησιμοποιηθεί για να μεταδώσουμε ένα μπλοκ από $k = \log_2 M$ bits ανά κυματομορφή σήματος. Σε αυτή την περίπτωση οι 8 κυματομορφές σήματος μπορούν να εκφρασθούν ως

$$u_m(t) = \sqrt{\frac{2E_S}{T}\cos(2\pi f_c t + 2\pi m\Delta f t)}, \qquad m = 0, 1, ..., 7 \tag{6}$$

όπου $E_s = kE_b$ είναι η ενέργεια ανά σύμβολο, $T = 8T_b$ η διάρκεια συμβόλου και Δf η συχνοτική απόσταση μεταξύ δύο διαδοχικών συχνοτήτων, δηλαδή, $\Delta = f_m - f_{m-1}$ όπου $f_m = f_c + m\Delta f$.

Οι M FSK κυματομορφές έχουν ίδια ενέργεια E_s. Η απόσταση συχνότητας Δf καθορίζει το βαθμό στον οποίο μπορούμε να διακρίνουμε μεταξύ τους τα M πιθανά μεταδιδόμενα σήματα.

Οι M-αδικές ορθογώνιες FSK κυματομορφές αναπαρίστανται γεωμετρικά ως M, M-διάστατα ορθογώνια διανύσματα, τα οποία δίνονται ως

$$\begin{aligned} s_1 &= (\sqrt{E_s}, 0, 0, ..., 0) \\ s_2 &= (0, 0, 0, ..., \sqrt{E_s}) \\ &... \\ s_M &= (0, 0, 0, ..., \sqrt{E_s}) \end{aligned}$$

όπου οι συναρτήσεις βάσης είναι $\psi_m(t) = \sqrt{\frac{2}{T}} \cos 2\pi(f_c + m\Delta f)t$. Η απόσταση μεταξύ δύο διανυσμάτων είναι $d = \sqrt{2E_s}$ για όλα τα m και n, που είναι και η ελάχιστη απόσταση μεταξύ των M σημάτων.

3.1 Αποδιαμόρφωση και Φώραση Σημάτων FSK

Τα FSK σήματα μεταδίδονται μέσω ενός AWGN καναλιού. Υποθέτουμε ότι το κάθε σήμα καθυστερεί κατά τη μετάδοση μέσα από το κανάλι.

Η αποδιαμόρφωση και φώραση των M-αδικών FSK σημάτων μπορεί να επιτευχθεί με δύο τρόπους. Μια τεχνική είναι να εκτιμήσουμε τις M ολισθήσεις φάσης $\{\phi_m\}$ [1] και να εκτελέσουμε αποδιαμόρφωση και φώραση σύμφωνης - φάσης (phase-coherent demodulation and detection). Η άλλη τεχνική είναι να αγνοηθούν οι φάσεις στην αποδιαμόρφωση και φώραση (non-coherent demodulation and detection).

Κατά την αποδιαμόρφωση σύμφωνης-φάσης, το λαμβανόμενο σήμα $r(t)$ συσχετίζεται με κάθε ένα από τα δυνατά M σήματα $\cos(2\pi f_c t + 2\pi m\Delta f t + \hat{\phi_m}), \quad m = 0, 1, ..., M-1$. Ένα διάγραμμα βαθμίδων αυτού του τύπου αποδιαμόρφωσης παρουσιάζεται στο στο σχήμα 4.

Αυτό τον τύπο αποδιαμόρφωσης χρησιμοποιήσαμε και στην άσκηση.

Αντίθετα, μια άλλη μέθοδος αποδιαμόρφωσης και φώρασης η οποία δεν απαιτεί γνώση των φάσεων φέροντος. Στην περίπτωση αυτή, υπάρχουν δύο συσχετιστές ανά κυματομορφή σήματος, δηλαδή, συνολικά $2M$ συσχετιστές. Το λαμβανόμενο σήμα συσχετίζεται με τις συναρτήσεις βάσης (ορθογώνια φέροντα) $\sqrt{\frac{2}{T}} \cos(2\pi f_c t + 2\pi m\Delta f t)$ και $\sqrt{\frac{2}{T}} \sin(2\pi f_c t + 2\pi m\Delta f t)$ για $m = 0, 1, ..., M-1$. Οι $2M$ έξοδοι των συσχετιστών δειγματοληπτούνται στο τέλος της περιόδου σηματοδοσίας, και τα δείγματα εισέρχονται στο φωρατή. Επομένως, εάν μεταδόθηκε το m-στο σήμα, τα $2M$ δείγματα στην k-στη είσοδο του φωρατή μπορούν να εκφρασθούν ως:

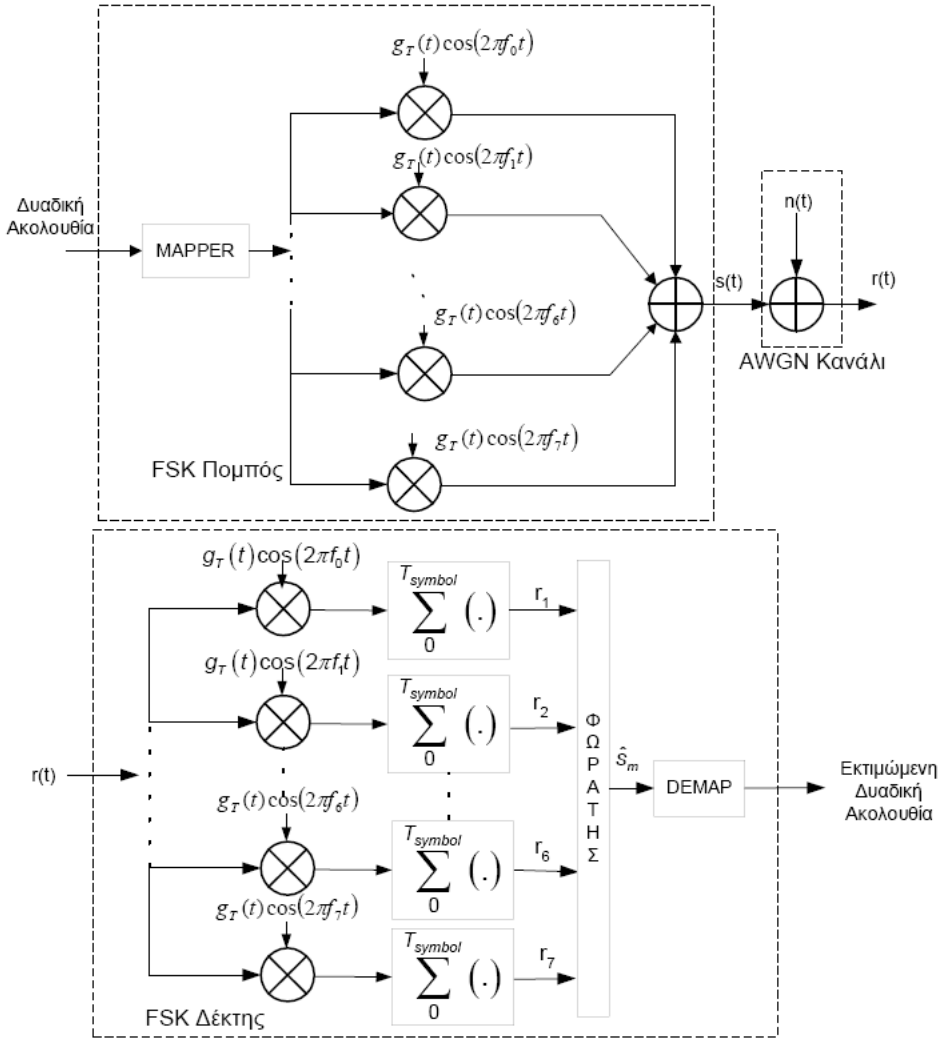
$$r_{kc} = \sqrt{E_s}[\frac{\sin 2\pi(k-m)\Delta fT}{2\pi(k-m)\Delta fT} \cos\phi_m - \frac{\cos 2\pi(k-m)\Delta fT - 1}{2\pi(k-m)\Delta fT} \sin\phi_m] + n_{kc}$$

$$r_{kc} = \sqrt{E_s}[\frac{\cos 2\pi(k-m)\Delta fT - 1}{2\pi(k-m)\Delta fT} \cos\phi_m - \frac{\sin 2\pi(k-m)\Delta fT}{2\pi(k-m)\Delta fT} \sin\phi_m] + n_{ks}$$

3.2 Κύκλωμα FSK

Στο σχήμα 4 απεικονίζεται ο σχεδιασμός ενός FSK κυκλώματος. Συγκεκριμένα μπορούμε να διακρίνουμε τα διάφορα modules που το απαρτίζουν.

[1] όπου ϕ_m δηλώνει την ολίσθηση φάσης του m-οστού σήματος (εξαιτίας της καθυστέρησης διάδοσης)



Σχήμα 4: Αναπαράσταση ομώδυνου 8-δικού FSK

Αρχικά, παράγεται η δυαδική ακολουθία η οποία διοχετεύεται στον mapper όπου και γίνεται η αντιστοίχηση της σε σύμβολα.

Η έξοδος του mapper διοχετεύεται με την σειρά της στον modulator ο οποίος διαμορφώνει την κάθε συνιστώσα. Το σύστημα 8-FSK που προσομοιώσαμε, χρησιμοποιεί τα εξής οκτώ σήματα για καθένα σύμβολο:

$$s_m(t) = g_T(t)\cos(2\pi(f_c + m\Delta f)t), \quad \Delta f = \frac{1}{T_{symbol}}, m = 0, ..., 7$$

οπότε οι 8 φέρουσες διαφέρουν κατά $\frac{1}{T_{symbol}}$ μεταξύ τους.

Στο σχήμα επίσης παρουσιάζεται και το κανάλι AWGN που πρόκειται για τον

θόρυβο που προκύπτει κατά την διάρκεια μετάδοσης του σήματος από τον πομπό του δέκτη. Στα πειράματα μας, ο θόρυβος είναι μηδενικής μέσης τιμής και διασποράς $\sigma = \frac{N_0}{2}$.

Με την σειρά του ο δέκτης, το λαμβανόμενο σήμα το αποδιαμορφώνει με τον demodulator. Εκεί γίνεται η αντίστοιχη αποδιαμόρφωση.

Στην συνέχεια ο φωρατής δέχεται ως είσοδο το διάνυσμα r και αποφασίζει σε ποιο σύμβολο βρίσκεται εγγύτερα. Το διάνυσμα s_m που θα έχει την μικρότερη απόσταση από το r αντιστοιχεί και στο σύμβολο που στάλθηκε (φωρατής μέγιστης πιθανοφάνειας). Συγκεκριμένα να πούμε στο σημείο αυτό πως αν π.χ. έχει σταλεί το σύμβολο s_1 (υποθέτουμε ένα 8-PSK) τότε τα $r_2, r_3, ..., r_8$ δε θα περιέχουν τίποτα άλλο παρά μόνο μία μικρή προσθήκη θορύβου. Με άλλα λόγια αν το r_i έχει τη μεγαλύτερη τιμή τότε το σύμβολο που θα έχει σταλεί θα είναι το i-οστό.

Τέλος, παιρνόντας στον demapper γίνεται η αντιστοίχηση στην εκτιμώμενη απεσταλθήσα δυαδική ακολουθία.

4 Πειραματική Διαδικασία και Αξιολόγηση

4.1 Βασικοί Ορισμοί

Σε αυτή τη φάση είμαστε έτοιμοι να αξιολογήσουμε τα συστήματα που περιγράψαμε παραπάνω και συγκεκριμένα να καταγράψουμε τα ακόλουθα:

- *Bit Error Rate (BER)*: η πιθανότητα εμφάνισης σφάλματος bit. Με άλλα λόγια να συγκρίνουμε κάθε λαμβανόμενο bit με κάθε bit που στάλθηκε από τον πομπό. Η συγκεκριμένη μετρική αναφέρεται προς το σύνολο των λανθασμένα απεσταλθέντων bit προς το συνολικό πλήθος απεσταλθέντων bit.

- *Symbol Error Rate (SER)*: δηλαδή τη πιθανότητα εμφάνισης σφάλματος συμβόλου. Με άλλα λόγια να συγκρίνουμε κάθε λαμβανόμενο σύμβολο με κάθε σύμβολο που στάλθηκε από τον πομπό. Η συγκεκριμένη μετρική αναφέρεται προς το σύνολο των λανθασμένα απεσταλθέντων symbols προς το συνολικό πλήθος απεσταλθέντων symbols.

Αξίζει να αναφέρουμε, όσον αφορά το πειραματικό μέρος της άσκησης, ότι οι παραπάνω αξιολογήσεις πραγματοποίηθηκαν σε σύστημα όπου γνωρίζαμε την πληροφορία η οποία αποστέλεται και αυτή που λαμβάνεται έτσι ώστε να μπορούμε να υπολογίσουμε επακριβώς τις συγκεκριμένες μετρικές. Προφανώς και κάτι τέτοιο δεν μπορεί να γίνει σε ένα πραγματικό περιβάλλον, γιατί αν ο δέκτης γνώριζε ήδη την πληροφορία που θα μεταδόσει ο πομπός, θα έπαυε αυτομάτως να υφίσταται ο λόγος μετάδοσης αυτής.

4.2 Περιγραφή Πειραματικών Κυκλωμάτων

Τα κυκλώματα στα οποία πραγματοποιήθηκε η εκτέλεση των ζητούμενων πειραμάτων περιγράφονται παρακάτω:

1. *8-PSK με κωδικοποίηση των συμβόλων κατά Gray*, για μεγαλύτερη ανοχή στα σφάλματα για τους λόγους που αναφέρθηκαν παραπάνω.

2. *8-FSK*

3. *8-PSK χωρίς κωδικοποίηση των συμβόλων κατά Gray*

Η μετρήσεις έγιναν για $SNR = [0 : 2 : 8]dB$ και για όγκο δεδομένων της τάξης των 10^5 bits για μεγαλύτερη αξιοπιστία στα παραγόμενα αποτελέσματα και συμπεράσματα. Συγκεκριμένα τα 10^5 bits μας προσφέρουν αξιοπιστία για μετρήσεις BER της τάξης του 10^{-3}, πράγμα που υπερκαλύπτει τις ανάγκες μας.

4.3 Benchmarking

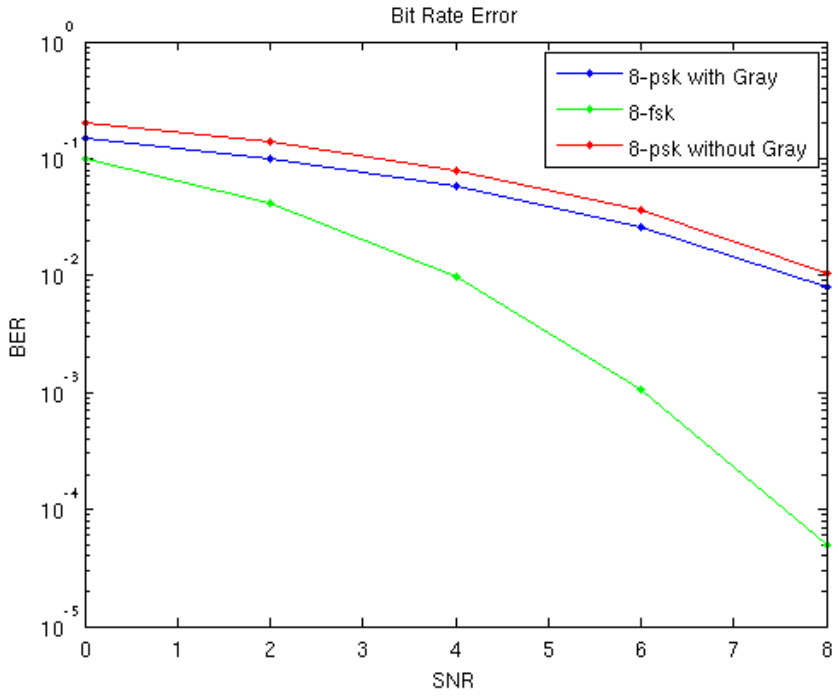
Ο κώδικας που χρησιμοποιήθηκε για την παραγωγή των παραπάνω μετρήσεων φαίνεται παρακάτω[2]:

```
bits = 10^5;

i = 1;
x = [0: 2: 8];

for SNR = 0: 2: 8
    A = binary_input(bits);
    B = mapper(A, 'psk', 1);
    C = modulator(B, 'psk');
    D = noise(C,SNR);
    r = demodulator(D, 'psk');
    E = foratis(r, 'psk');
    F = demapper(E', 'psk', 1);
    BER_psk_gray(i, 1) = ber(A,F);
    SER_psk_gray(i, 1) = ser(A,F);

    A = binary_input(bits);
    B = mapper(A, 'fsk', 0);
    C = modulator(B, 'fsk');
    D = noise(C,SNR);
    r = demodulator(D, 'fsk');
    E = foratis(r, 'fsk');
    F = demapper(E', 'fsk', 0);
    BER_fsk(i, 1) = ber(A,F);
    SER_fsk(i, 1) = ser(A,F);

    A = binary_input(bits);
    B = mapper(A, 'psk', 0);
    C = modulator(B, 'psk');
    D = noise(C,SNR);
    r = demodulator(D, 'psk');
    E = foratis(r, 'psk');
    F = demapper(E', 'psk', 0);
```

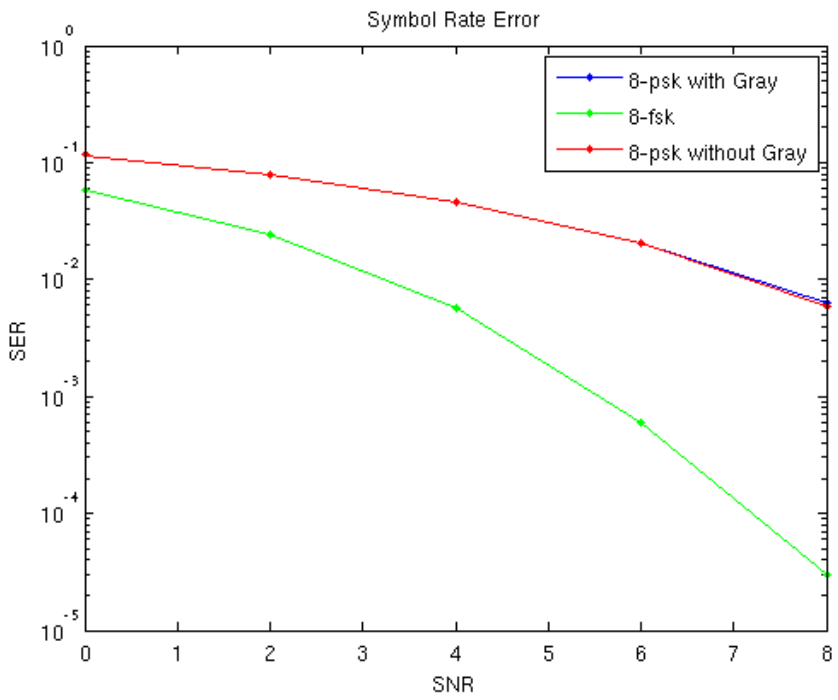
[2]Για τις αντίστοιχες συναρτήσεις που χρησιμοποιούνται μπορείτε να ανατρέξετε στο παράρτημα.

```
    BER_psk_without_gray(i, 1) = ber(A,F);
    SER_psk_without_gray(i, 1) = ser(A,F);


    i = i + 1;
end

semilogy(x', BER_psk_gray, '.-');
hold on;
semilogy(x', BER_fsk, 'g.-');
semilogy(x', BER_psk_without_gray, 'r.-');
legend('8-psk with Gray','8-fsk','8-psk without Gray');
title('Bit Rate Error');
xlabel('SNR');
ylabel('BER');
hold;

figure;

semilogy(x', SER_psk_gray, '.-');
hold on;
semilogy(x', SER_fsk, 'g.-');
semilogy(x', SER_psk_without_gray, 'r.-');
legend('8-psk with Gray','8-fsk','8-psk without Gray');
title('Symbol Rate Error');
xlabel('SNR');
ylabel('SER');
hold;
```

Τα αποτελέσματα που λάβαμε απεικονίζονται στις δύο γραφικές παραστάσεις που φαίνονται στα σχήματα 5 και 6



Σχήμα 5: Bit Error Rate (BER)



Σχήμα 6: Symbol Error Rate (SER)

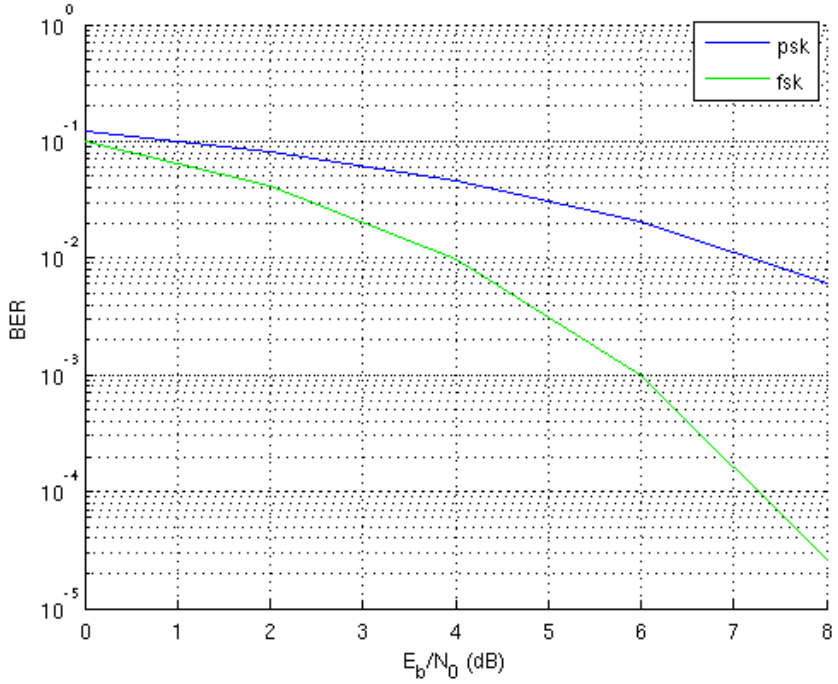
Επειδή τα πιο πιθανά σφάλματα οδηγούν στην εσφαλμένη επιλογή μιας γειτονικής φάσης ως προς τη πραγματική, τα περισσότερα εσφαλμένα k-bit σύμβολα περιέχουν μόνο ένα εσφαλμένο bit. Έτσι το σύστημα 8-PSK με κωδικοποίηση των συμβόλων κατά Gray είναι λογικό να επηρεάζεται λιγότερο από πιθανά σφάλματα, στη περίπτωση που μελετάμε την πιθανότητα εμφάνισης σφάλματος bit, σε σχέση με το 8-PSK χωρίς κωδικοποίηση των συμβόλων κατά Gray. Αυτό συμβαίνει γιατί με την χρήση κωδικοποίησης κατά Gray δυο γειτονικά k-bit σύμβολα διαφέρουν μόνο κατά ένα bit, με συνέπεια, αν αναγνωρίσουμε λάθος σύμβολο, το οποίο με μεγάλη πιθανότητα θα είναι γειτονικό ως προς το ορθό, και επομένως αυτά τα δύο να διαφέρουν μόνο κατά 1 bit μεταξύ τους.

Παρόλα αυτά ο Gray δε μπορεί να μας διασφαλίσει το ίδιο και στη περίπτωση που μελετάμε τη πιθανότητα εμφάνισης σφάλματος συμβόλου. Με άλλα λόγια και στα δύο PSK η πιθανότητα εμφάνισης κάποιου σφάλματος συμβόλου θα είναι ακριβώς η ίδια πράγμα που επιβεβαιώνεται κιόλας από την αντίστοιχη γραφική παράσταση.

Όσον αφορά τώρα τη σύγκριση του 8-FSK με τα 8-PSK φαίνεται ότι στη περίπτωση του BER το 8-FSK παρουσιάζει πολύ καλύτερη ανοχή στα σφάλματα από αυτό των 8-PSK. *Τα FSK συστήματα παρουσιάζουν καλύτερη συμπεριφορά στο θόρυβο σε σχέση με τα PSK αλλά είναι και πιο περίπλοκα.*

Να πούμε εδώ ότι η Matlab διαθέτει ένα εργαλείο που μας επιτρέπει να σχεδιάσουμε άμεσα και να συγκρίνουμε τέτοια συστήματα για διάφορους τύπους

θορύβων και M τα Bit Error Rates, το *bertool*. Το εργαλείο αυτό μας βοήθησε να επιβεβαιώσουμε την ορθότητα των μετρήσεών μας. Ένα παράδειγμα αυτού φαίνεται στο σχήμα 7, για $M = 8$.



Σχήμα 7: Bit Error Rate (BER) με τη χρήση της bertool

Κλείνοντας την ενότητα αυτή τον πειραματικών αναλύσεων να πούμε ότι όσο αυξάνεται το SNR που παρέχουμε στο 8-FSK τόσο περισσότερο SNR απαιτεί ένα σύστημα 8-PSK με ή χωρίς κωδικοποίηση των συμβόλων κατά Gray ώστε να παρέχει την ίδια μικρή πιθανότητα εμφάνισης σφάλματος bit.

5 Σχολιασμός και Σύγκριση Διαμορφώσεων M-PSK και M-FSK

Στην ενότητα αυτή καλούμαστε αρχικά να σχολιάσουμε τα σχήματα 7.57 και 7.63 από το [1] και στη συνέχεια να σχολιάσουμε και να συγκρίνουμε τις διαμορφώσεις M-PSK, M-FSK, ως προς το ρυθμό μετάδοσης bits, την πιθανότητα σφάλματος, και το απαιτούμενο εύρος ζώνης όταν αυξάνει το M.

Στο σχήμα 7.57 φαίνεται η πιθανότητα του σφάλματος συμβόλου για κάποιο M-PSK συναρτήσει του SNR/bit για $M = 2, 4, 6, 8, 16$ και 32. Από τα διάφορα γραφήματα μπορούμε να καταλάβουμε ότι όταν για κάποιο συγκεκριμένο M το SNR/bit μεγαλώνει το SER γίνεται όλο και πιο μικρό τείνοντας στο τέλος σε τιμές

τις τάξης του 10^{-5}. Επίσης όσο το M μεγαλώνει τόσο μεγαλύτερα SNR/bit dB χρειάζονται για την παραγωγή ίδιας πιθανότητας σφαλμάτων συμβόλου. Συγκεκριμένα όταν $P_M = 10^{-5}$, η διαφορά επίδοσης μεταξύ του $M = 4$ και $M = 8$ είναι περίπου $4dB$, και η διαφορά μεταξύ του $M = 8$ και $M = 16$ είναι περίπου $5dB$. Για μεγάλες τιμές του M, ο διπλασιασμός του αριθμού των φάσεων απαιτεί επιπρόσθετα $6dB/bit$ για την επίτευξη της ίδιας επίδοσης. Μια προσέγγιση της πιθανότητας σφάλματος για μεγάλες τιμές του M και του SNR, για $\frac{E_s}{N_0} \gg 1$ και $\mid \Theta_r \mid \preceq \frac{\pi}{2}$ είναι η ακόλουθη:

$$P_M \approx 2Q(\sqrt{2\varrho_s} sin\frac{\pi}{M}) \text{ όπου } M = 2_k \text{ και } \varrho_s = k\varrho_b \tag{7}$$

Η ισοδύναμη πιθανότητα σφάλματος bit στη M-αδική διαμόρφωση κατά φάση με κωδικοποίηση κατά Gray, προσεγγίζεται πολύ καλά ως:

$$P_b \approx \frac{1}{k} P_M \tag{8}$$

Όσον αφορά το σχήμα 7.63 τώρα, αναπαριστά την πιθανότητα σφάλματος bit για κάποιο M-FSK συναρτήσει του SNR/bit για $M = 2, 4, 8, 16, 32$ και 64. Από τα διάφορα γραφήματα μπορούμε να καταλάβουμε ότι όταν για κάποιο συγκεκριμένο M το SNR/bit μεγαλώνει το BER γίνεται όλο και πιο μικρό τείνοντας στο τέλος σε τιμές τις τάξης του 10^{-6}. Επίσης όσο το M μεγαλώνει τόσο μικρότερα SNR/bit dB χρειάζονται για την παραγωγή ίδιας πιθανότητας σφαλμάτων συμβόλου. Συγκεκριμένα για να επιτύχουμε $P_b = 10^{-5}$, το απαιτούμενο SNR/bit είναι λίγο περισσότερο των $12dB$ για $M = 2$, αλλ·α αν το M αυξηθεί σε 64 ($k = 6\frac{bits}{\text{σύμβολο}}$), το απαιτούμενο SNR/bit είναι περίπου $6dB$. Έτσι αυξάνοντας το M από 2 σε 64, εξοικονομούμε περισσότερο από $6dB$ (ένα παράγοντα ελάττωσης ίσο με 4) στη μεταδιδόμενη ισχύ (ή ενέργεια) που απαιτείται για να επιτύχουμε $P_b = 10^{-5}$.

Κλείνοντας και αυτή την ενότητα, συνοψίζουμε τα εξής σε σχέση με τα *M-PSK* και *M-FSK*:

- *M-PSK*:
  1. υψηλό ρυθμό μετάδοσης bits
  2. μεγάλη πιθανότητα σφάλματος
  3. όλο και μεγαλύτερο εύρος ζώνης όσο αυξάνει το M
- *M-FSK*:

1. χαμηλό ρυθμό μετάδοσης bits
2. μικρή πιθανότητα σφάλματος
3. όλο και μικρότερο εύρος ζώνης όσο αυξάνει το M

6 Εργαλεία ανάπτυξης

Η συγγραφή και ανάπτυξη του κώδικα έγινε σε GNU/Linux, Debian Lenny . Χρησιμοποιήθηκαν:

- **Matlab R2008a** (7.6.0.324)
- **LaTeX** για την συγγραφή της αναφοράς.

7 Παράρτημα

7.1 Παράθεση κώδικα Matlab

7.1.1 Binary Input

```
function binary_sequence = binary_input(number_of_elements)
% binary_sequence = binary_input(number_of_elements):
%
% The binary_input is a function that takes as argument the number of
% elements (0, 1) that will be exported in the binary_sequence array

% initialize the binary_sequence with 0 or 1 as elements, that will have the
% same propability
binary_sequence = randsrc(number_of_elements, 1, [0,1]);

end
```

7.1.2 Mapper

```
function symbols_array = mapper(binary_sequence, encoding, gray)
% s_m = mapper_modulator(binary_sequence, encoding, gray):
%
% The mapper_modulator is a function that take as argument a binary
% sequence and transforms the elements of this array into symbols
% The encoding is either 'fsk' or 'psk'
% The gray argument denotes if is to be used gray (1) encoding or not (0)

% the length of input
size_of_binary_sequence = length(binary_sequence);

% we group the bits into groups of 3
% the remainder of the sequence is separately converted into one symbol at
% the end
temp = mod(size_of_binary_sequence, 3);

% the sequence which is dividable by 3
new_bin_seq = binary_sequence(1 : (size_of_binary_sequence - temp), :);
```

```
% grouping of that sequence
reshaped_sequence = reshape(new_bin_seq, 3, (size_of_binary_sequence - temp) / 3);

% tranform the sequence into binary code for every group of 3 bits
for i = 1: (size_of_binary_sequence - temp) / 3
    symbols_array(i) = bin2dec(num2str(reshaped_sequence(:, i)'));
end

% the rest of the bits are separately tranformed into a symbol in binary
% code
if temp ~= 0
    symbols_array(i + 1) = bin2dec(num2str(binary_sequence(size_of_binary_sequence - temp +
                                        + 1 :size_of_binary_sequence, 1)'));
end

% if we use gray encoding in order to achieve smaller distance among two
% symbols which are adjacent, we encode the symbols into Gray by using the
% following function bin2gray
if gray == 1
    symbols_array = bin2gray(symbols_array, encoding, 8);
end
```

7.1.3 Modulator

```
function s_m = modulator(symbols_array, encoding)
% modulator(symbols_array, encoding)
%
% Takes as arguments the symbols array that are to be transmitted and
% encodes it according to the argument encoding ('psk' or 'fsk')
% Returns the modulated signal

% size of the array that has the sequence converted into symbols
size_of_symbols_array = length(symbols_array);

% period of symbol
T_symbol = 40;
% frequency of symbol
f_symbol = 1 / T_symbol;
% period of sample
T_sample = 1;
% period of ferousa
T_c = 4;
% frequency of ferousa
f_c = 1 / T_c;
% Energy per symbol
E_s = 1;

% orthogonal pulse
g = sqrt(2 * E_s / T_symbol);

% initialization of the symbols that we send
s_m = zeros(size_of_symbols_array, T_symbol / T_sample);

% computation of the transmitted signal
if encoding == 'psk'
    for i = 1: size_of_symbols_array
        for t = 1: T_symbol/T_sample
            s_m(i, t) = g * cos( 2*pi*f_c*t - 2*pi*symbols_array(i)/8 );
        end
    end
elseif encoding == 'fsk'
    for i = 1: size_of_symbols_array
```

```
        for t = 1: T_symbol/T_sample
            s_m(i, t) = g * cos(2 * pi * (f_c + symbols_array(i) * f_symbol) * t);
        end
    end
end
```

7.1.4 Προσθήκη Θορύβου

```
function received_signal = noise(s_m, SNR)
% received_signal = noise(s_m)
% The noise function takes as argument the s_m signal that is to be
% transmitted and the SNR and adds AWGN

% we solve the equation
% 10 * log_10(E_b / N_0) = SNR
% and try to find N_0
% given that
E_s = 1; % and
E_b = E_s / 3;
% we have as a result
N_0 = E_b / (10^(SNR/10));

% We create a gaussian distribution with mean value:
m = 0;
% and standard deviation
sigma = sqrt(N_0 / 2);

% the noise is added to every sample taken by the modulator
% for that reason, the derived array has to have the same dimensions as the
% array of the samples
[L_symbol, T_symbol] = size(s_m);

% produce AWGN
noise = m + sigma * randn(L_symbol, T_symbol);

% adds it to the signal
received_signal = s_m + noise;
```

7.1.5 Demodulator

```
function r = demodulator(received_signal, encoding)
% [r1, r2] = demodulator(received_signal)
% The demodulator function takes as argument the received signal and finds
% the components (r1, ...) of every transmitted signal
% The encoding could be either 'psk' or 'fsk'

% period of symbol
T_symbol = 40;
% frequency of symbol
f_symbol = 1 / T_symbol;
% period of sample
T_sample = 1;
% period of ferousa
T_c = 4;
% frequency of ferousa
f_c = 1 / T_c;
% Energy per symbol
```

```
E_s = 1;

% orthogonal pulse
g = sqrt(2 * E_s / T_symbol);

[L_symbol, T_symbol] = size(received_signal);

% demodulation
if encoding == 'psk'
    for t = 1: T_symbol
        y1(t, 1) = g * cos(2 * pi * f_c * t);
        y2(t, 1) = g * sin(2 * pi * f_c * t);
    end

    % calculation of the 2 components
    r = [received_signal * y1, received_signal * y2];
elseif encoding == 'fsk'
    for i = 1: 8
        for t = 1: T_symbol
            y(i, t) = g * cos(2 * pi * ( f_c + (i - 1) * f_symbol) * t);
        end
    end

    % calculation of the 8 components
    r = received_signal * y';
end
```

7.1.6 Φωρατής

```
function symbols = foratis(r, encoding)
% symbols = foratis(r1, r2)
% The foratis function takes the r argument and calculates the
% binary (or gray) symbols that was to be send
% The encoding could be either 'psk' or 'fsk'

[r_lines, r_columns] = size(r);

if encoding == 'psk'
    % calculates each possible received symbol
    for i = 1: 8
        s(i, 1) = cos( 2 * pi * i / 8 );
        s(i, 2) = sin( 2 * pi * i / 8 );
    end

    % calculates the symbol which presents the greatest propability to
    % be the sent symbol
    for j =1: r_lines
        for i = 1: 8
            temp(i, 1) = norm([r(j,1), r(j,2)] - s(i,:));
        end
        [min_diff, symbols(j, 1)] = min(temp);
    end

    % the 8th symbol is actually the 0th symbol
    symbols = mod(symbols,8);

elseif encoding == 'fsk'
    % period of symbol
    T_symbol = 40;
    % frequency of symbol
    f_symbol = 1 / T_symbol;
```

```
    % calculates the symbol which presents the greatest probability to
    % be the sent symbol
    for j =1: r_lines
        [max_diff, symbols(j,1)] = max( r(j, :) );
    end

    symbols = symbols - 1;
end
```

7.1.7 Demapper

```
function received_bits = demapper(symbols, encoding, gray)
% received_bits = demapper(symbols)
% The demapper function converts the received symbols to bits
% The encoding could be either 'psk' or 'fsk' and also with gray encoding
% (1) or not (0)

% if there has been used Gray encoding in the transmitted signal
if gray == 1
    symbols = gray2bin(symbols, encoding, 8);
end

received_bits = dec2bin(symbols);

% m: number of lines of the received bits matrix
% n: number of columns of the received bits matrix
[m, n] = size(received_bits);

% reshape the matrix with the received bits to an array
received_bits = reshape(received_bits', m*n, 1);

% convert to double every character
% in orde to recover the value that this character represents in ASCII code
% we substruct 30(hex) = 48(dec)
% we have assumed that we deal only with character wich represents digits
% that is a valid hypothesis because we deal only with zero and one
received_bits = double(received_bits) - 48;
```

References

[1] Masoud Salehi Proakis, John G. *Communication systems engineering*. Prentice Hall, 2002.