ECS455: Chapter 4

Multiple Access

4.7 Synchronous CDMA

Dr.Prapun Suksompong
prapun.com/ecs455

Office Hours:
BKD 3601-7

Tuesday	**9:30-10:30**
Tuesday	**13:30-14:30**
Thursday	**13:30-14:30**

Synchronous CDMA Model

- Timing is important for orthogonality
- It is not possible to obtain orthogonal codes for asynchronous users. [Goldsmith, 2005, Sec. 13.4, p. 425]
- Bit epochs are aligned at the receiver [Verdu, 1998, p 21]
- Require
  - Closed-loop timing control or
  - Providing the transmitters with access to a common clock (such as the Global Positioning System) [Verdu, 1998, p 21]

Walsh Functions [Walsh, 1923]

- Used in second- (2G) and third-generation (3G) cellular radio systems for providing channelization
- A set of Walsh functions can be **ordered** according to the number of **zero crossing** (sign changes)

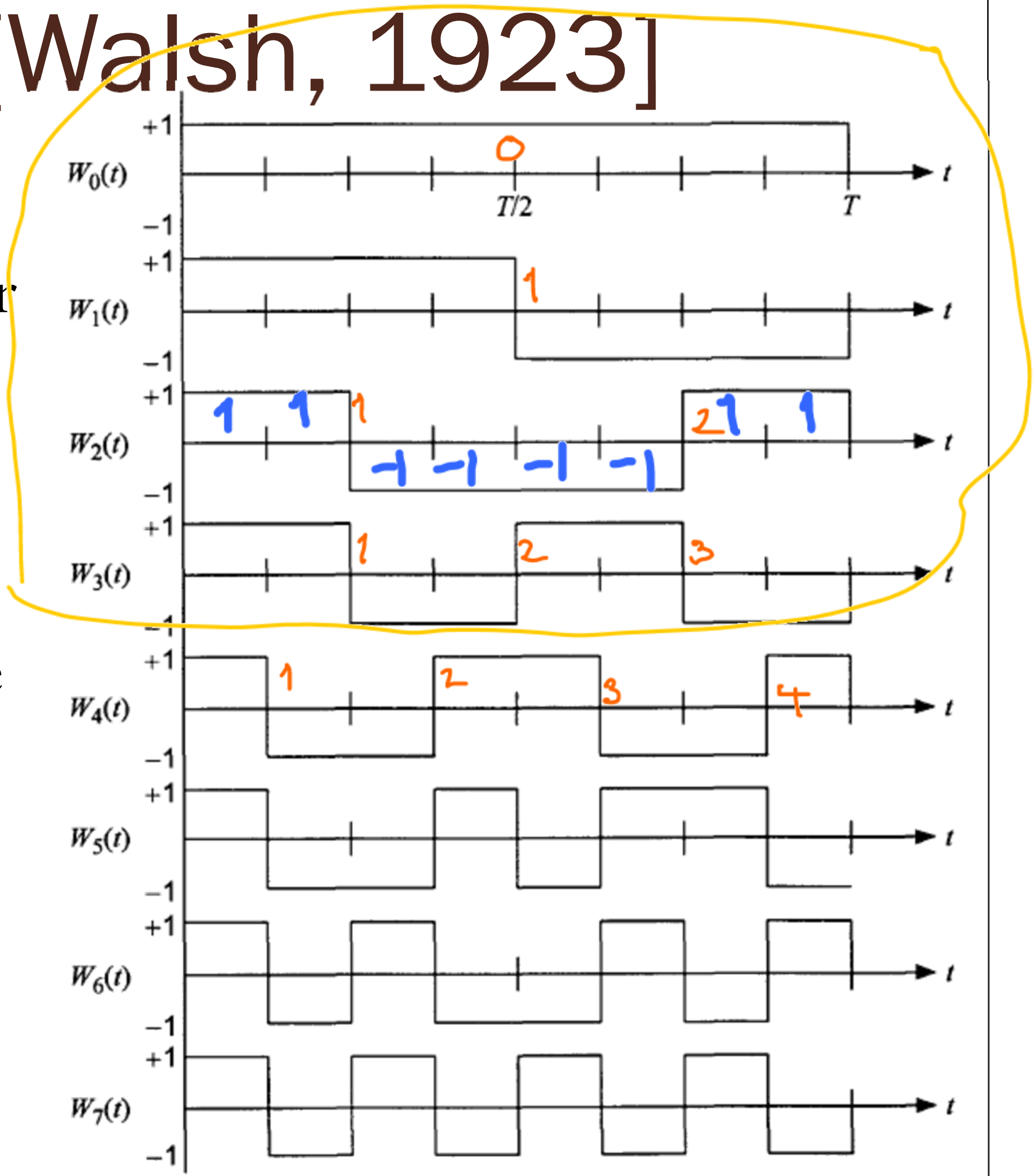


Figure 5.1 The Walsh functions of order 8. [Lee and Miller, 1998, Fig. 5.1]

Walsh Functions (2)

We define the Walsh functions of order N as a set of N time functions, denoted $\{W_j(t);\ t \in (0, T),\ j = 0, 1, \ldots, N-1\}$, such that

- $W_j(t)$ takes on the values $\{+1, -1\}$ except at the jumps, where it takes the value zero.
- $W_j(0) = 1$ for all j.
- $W_j(t)$ has precisely j sign changes (zero crossings) in the interval $(0, T)$.
- $$\int_0^T W_j(t)\, W_k(t)\, dt = \begin{cases} 0, & \text{if } j \neq k \\ T, & \text{if } j = k \end{cases}$$ Orthogonality
- Each function $W_j(t)$ is either odd or even with respect to the mid-point of the interval.

Application:

Once we know how to generate these Walsh functions of any order N, we can use them in N-channel orthogonal multiplexing applications.

Walsh Sequences

Walsh sequences
$W_0 = 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0$
$W_1 = 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1$
$W_2 = 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0$
$W_3 = 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1$
$W_4 = 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0$
$W_5 = 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1$
$W_6 = 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0$
$W_7 = 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1$
$W_8 = 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0$
$W_9 = 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1$
$W_{10} = 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0$
$W_{11} = 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1$
$W_{12} = 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0$
$W_{13} = 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1$
$W_{14} = 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0$
$W_{15} = 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1$

- The Walsh functions, expressed in terms of {+1,-1} values, form a group under the multiplication operation (**multiplicative group**).
- The Walsh sequences, expressed in terms of {0, 1} values, form a group under modulo-2 addition (**additive group**).
- **Closure property**:

$$W_i(t) \cdot W_j(t) = W_r(t)$$

$$W_i \oplus W_j = W_r$$

Abstract Algebra

- A ***group*** is a set of objects G on which a binary operation "$\cdot$" has been defined. $"\cdot": G \times G \rightarrow G$ (closure). The operation must also satisfy
  1. Associativity: $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
  2. Identity: $\exists e \in G$ such that $\forall a \in G \quad a \cdot e = e \cdot a = a \quad \exists a \in G$
  3. Inverse: $\forall a \in G \quad \exists$ a unique element $a^{-1} \in G$ such that $a \cdot a^{-1} = a^{-1} \cdot a = e$.
- A group is said to be ***commutative*** (or ***abelian***) if it also satisfies commutativity:

$$\forall a, b \in G \,,\; a \cdot b = b \cdot a \,.$$

- The group operation for a commutative group is usually represented using the symbol "+", and the group is sometimes said to be "additive."

Walsh sequences of order 64

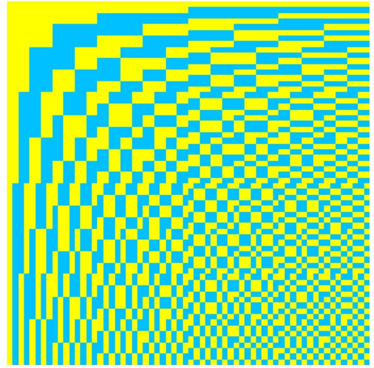


Table 5.2 Walsh functions of order 64 (indexed by zero crossings)

W_0	0000000000000000	0000000000000000	0000000000000000	0000000000000000
W_1	0000000000000000	0000000000000000	1111111111111111	1111111111111111
W_2	0000000000000000	1111111111111111	1111111111111111	0000000000000000
W_3	0000000000000000	1111111111111111	0000000000000000	1111111111111111
W_4	0000000011111111	1111111100000000	0000000011111111	1111111100000000
W_5	0000000011111111	1111111100000000	1111111100000000	0000000011111111
W_6	0000000011111111	0000000011111111	1111111100000000	1111111100000000
W_7	0000000011111111	0000000011111111	0000000011111111	0000000011111111
W_8	0000111111110000	0000111111110000	0000111111110000	0000111111110000
W_9	0000111111110000	0000111111110000	1111000000001111	1111000000001111
W_{10}	0000111111110000	1111000000001111	1111000000001111	0000111111110000
W_{11}	0000111111110000	1111000000001111	0000111111110000	1111000000001111
W_{12}	0000111100001111	1111000011110000	0000111100001111	1111000011110000
W_{13}	0000111100001111	1111000011110000	1111000011110000	0000111100001111
W_{14}	0000111100001111	0000111100001111	1111000011110000	1111000011110000
W_{15}	0000111100001111	0000111100001111	0000111100001111	0000111100001111
W_{16}	0011110000111100	0011110000111100	0011110000111100	0011110000111100
W_{17}	0011110000111100	0011110000111100	1100001111000011	1100001111000011
W_{18}	0011110000111100	1100001111000011	1100001111000011	0011110000111100
W_{19}	0011110000111100	1100001111000011	0011110000111100	1100001111000011
W_{20}	0011110011000011	1100001100111100	0011110011000011	1100001100111100
W_{21}	0011110011000011	1100001100111100	1100001100111100	0011110011000011
W_{22}	0011110011000011	0011110011000011	1100001100111100	1100001100111100
W_{23}	0011110011000011	0011110011000011	0011110011000011	0011110011000011
W_{24}	0011001111001100	0011001111001100	0011001111001100	0011001111001100
W_{25}	0011001111001100	0011001111001100	1100110000110011	1100110000110011
W_{26}	0011001111001100	1100110000110011	1100110000110011	0011001111001100
W_{27}	0011001111001100	1100110000110011	0011001111001100	1100110000110011
W_{28}	0011001100110011	1100110011001100	0011001100110011	1100110011001100
W_{29}	0011001100110011	1100110011001100	1100110011001100	0011001100110011
W_{30}	0011001100110011	0011001100110011	1100110011001100	1100110011001100
W_{31}	0011001100110011	0011001100110011	0011001100110011	0011001100110011
W_{32}	0110011001100110	0110011001100110	0110011001100110	0110011001100110
W_{33}	0110011001100110	0110011001100110	1001100110011001	1001100110011001
W_{34}	0110011001100110	1001100110011001	1001100110011001	0110011001100110
W_{35}	0110011001100110	1001100110011001	0110011001100110	1001100110011001
W_{36}	0110011010011001	1001100101100110	0110011010011001	1001100101100110
W_{37}	0110011010011001	1001100101100110	1001100101100110	0110011010011001
W_{38}	0110011010011001	0110011010011001	1001100101100110	1001100101100110
W_{39}	0110011010011001	0110011010011001	0110011010011001	0110011010011001
W_{40}	0110100110010110	0110100110010110	0110100110010110	0110100110010110
W_{41}	0110100110010110	0110100110010110	1001011001101001	1001011001101001
W_{43}	0110100110010110	1001011001101001	0110100110010110	1001011001101001
W_{44}	0110100101101001	1001011010010110	0110100101101001	1001011010010110
W_{45}	0110100101101001	1001011010010110	1001011010010110	0110100101101001
W_{46}	0110100101101001	0110100101101001	1001011010010110	1001011010010110
W_{47}	0110100101101001	0110100101101001	0110100101101001	0110100101101001
W_{48}	0101101001011010	0101101001011010	0101101001011010	0101101001011010
W_{49}	0101101001011010	0101101001011010	1010010110100101	1010010110100101
W_{50}	0101101001011010	1010010110100101	1010010110100101	0101101001011010
W_{51}	0101101001011010	1010010110100101	0101101001011010	1010010110100101
W_{52}	0101101010100101	1010010101011010	0101101010100101	1010010101011010
W_{53}	0101101010100101	1010010101011010	1010010101011010	0101101010100101
W_{54}	0101101010100101	0101101010100101	1010010101011010	1010010101011010
W_{55}	0101101010100101	0101101010100101	0101101010100101	0101101010100101
W_{56}	0101010110101010	0101010110101010	0101010110101010	0101010110101010
W_{57}	0101010110101010	0101010110101010	1010101001010101	1010101001010101
W_{58}	0101010110101010	1010101001010101	1010101001010101	0101010110101010
W_{59}	0101010110101010	1010101001010101	0101010110101010	1010101001010101
W_{60}	0101010101010101	1010101010101010	0101010101010101	1010101010101010
W_{61}	0101010101010101	1010101010101010	1010101010101010	0101010101010101
W_{62}	0101010101010101	0101010101010101	1010101010101010	1010101010101010
W_{63}	0101010101010101	0101010101010101	0101010101010101	0101010101010101

What's wrong with this list?!

[Lee and Miller, 1998, Table 5.2]

Walsh Function Generation

- We can construct the Walsh functions by:
  1. Using Rademacher functions
  2. Using **Hadamard matrices**
  3. Exploiting the symmetry properties of Walsh functions themselves
- The **Hadamard matrix** is a square array of plus and minus ones, **{+1, -1}**, whose rows and columns are mutually orthogonal.
- If the first row and first column contain only plus ones, the matrix is said to be in **normal form**.
- We can replace "+1" with "0" and "-1" with "1" to express the Hadamard matrix using the logic elements **{0, 1}**.
- The 2×2 Hadamard matrix of order 2 is

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \equiv \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

(Named after the French mathematician Jacques Hadamard)

Hadamard matrix (1)

Suppose H_N is an $N \times N$ Hadamard matrix. $N \geq 1$ is called the order of a Hadamard matrix

1. $N = 1$, 2, or $4t$, where t is a positive integer.
2. $H_N H_N^T = NI_N$ where I_N is the $N \times N$ identity matrix.
3. If H_a and H_b are Hadamard matrices of order a and b, respectively, then we define $H_a \otimes H_b$ to be the Hadamard matrix H_{ab} of order ab whose elements are found by substituting H_b for $+1$ (or logic 0) in H_a and $-H_b$ (or the complement of H_b) for -1 (or logic 1) in H_a.

Caution: Some textbooks write this symbol as ×. It is not the regular matrix multiplication

Kronecker Product

- An operation on two matrices of arbitrary size
- Named after German mathematician Leopold Kronecker.
- If **A** is an *m*-by-*n* matrix and **B** is a *p*-by-*q* matrix, then the **Kronecker product** **A**⊗**B** is the *mp*-by-*nq* matrix

$$\mathbf{A}\otimes\mathbf{B}=\begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}.$$

- Example

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}\otimes\begin{bmatrix} 0 & 5 \\ 6 & 7 \end{bmatrix}=\begin{bmatrix} 1\cdot 0 & 1\cdot 5 & 2\cdot 0 & 2\cdot 5 \\ 1\cdot 6 & 1\cdot 7 & 2\cdot 6 & 2\cdot 7 \\ 3\cdot 0 & 3\cdot 5 & 4\cdot 0 & 4\cdot 5 \\ 3\cdot 6 & 3\cdot 7 & 4\cdot 6 & 4\cdot 7 \end{bmatrix}=\begin{bmatrix} 0 & 5 & 0 & 10 \\ 6 & 7 & 12 & 14 \\ 0 & 15 & 0 & 20 \\ 18 & 21 & 24 & 28 \end{bmatrix}.$$

A
B
2×B
3×B

Hadamard matrix (2)

- Consequently, if N is a power of two and it is understood that $H_1 = [+1] \equiv [0]$, then H_{2N} can be found as follows:

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \otimes H_N = H_2 \otimes H_N = H_{2N} = \begin{bmatrix} H_N & H_N \\ H_N & \overline{H_N} \end{bmatrix}$$

  where $\overline{H_N}$ is the negative (complement) of H_N.

- Hadamard matrices of order $N = 2^t$ can be formed by repeatedly multiplying ($\otimes$) the normal form of the $N = 2$ Hadamard matrix by itself.

Hadamard matrix: Examples

$$\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$H_4 = H_{2\times 2} = \begin{bmatrix} H_2 & H_2 \\ H_2 & \overline{H}_2 \end{bmatrix}$

$$\mathbf{H}_4 = \mathbf{H}_2 \otimes \mathbf{H}_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

$H_8 = H_{2\times 4}$

$$\mathbf{H}_8 = \mathbf{H}_2 \otimes \mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

$H_4 \quad H_4 \quad H_4 \quad \overline{H}_4$

$$\mathbf{H}_{16} = \mathbf{H}_2 \otimes \mathbf{H}_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 \end{bmatrix}$$

In MATLAB, use

```
hadamard(k)
```

Two ways to get H_8 from H_2 and H_4

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad H_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

$$H_8 = H_2 \otimes H_4$$

$$\left[\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right]$$

$$H_8 = H_4 \otimes H_2$$

$$\left[\begin{array}{cc|cc|cc|cc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ \hline 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ \hline 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right]$$

$$H_{16} = H_2 \otimes H_8 = H_8 \otimes H_2 = H_4 \otimes H_4$$

Properties

- Orthogonality:
  - Geometric interpretation: every two different rows represent two perpendicular vectors
  - Combinatorial interpretation: every two different rows have matching entries in exactly half of their elements and mismatched entries in the remaining elements.
- Symmetric
- Closure property
- The elements in the first column and the first row are all 1s. The elements in all the other rows and columns are evenly divided between 1 and -1.
- Traceless property

$tr(A)$ = sum of the main diagonal elements

$tr(H_N) = 0$

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

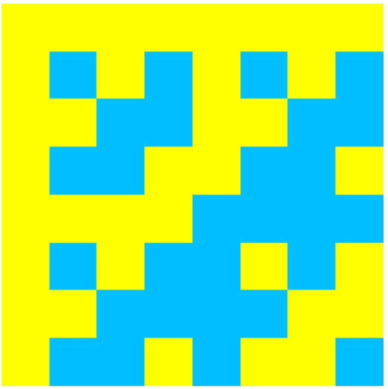
Walsh–Hadamard Sequences

- All the rows (or columns) of Hadamard matrices are Walsh sequences if the order is $N = 2^t$.
- Rows of the Hadamard matrix are *not indexed according to the number of sign changes*.
- Used in synchronous CDMA
  - It is possible to synchronize users on the downlink, where all signals originate from the same transmitter.
  - It is more challenging to synchronize users in the uplink, since they are not co-located.
    - Asynchronous CDMA

Hadamard Matrix in MATLAB

- We use the `hadamard` function in MATLAB to generate Hadamard matrix.

```
N = 8;  % Length of Walsh (Hadamard) functions
hadamardMatrix = hadamard(N)
hadamardMatrix =
```



1	1	1	1	1	1	1	1
1	-1	1	-1	1	-1	1	-1
1	1	-1	-1	1	1	-1	-1
1	-1	-1	1	1	-1	-1	1
1	1	1	1	-1	-1	-1	-1
1	-1	1	-1	-1	1	-1	1
1	1	-1	-1	-1	-1	1	1
1	-1	-1	1	-1	1	1	-1

- The Walsh functions in the matrix are not arranged in increasing order of their sequencies or number of zero-crossings (i.e. 'sequency order') .

Walsh Matrix in MATLAB

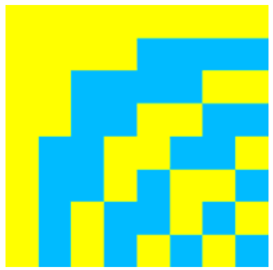
- The Walsh matrix, which contains the Walsh functions along the rows or columns in the increasing order of their sequencies is obtained by changing the index of the `hadamardMatrix` as follows.

```
HadIdx = 0:N-1;                          % Hadamard index
M = log2(N)+1;                           % Number of bits to represent the index
```

- Each column of the sequency index (in binary format) is given by the modulo-2 addition of columns of the bit-reversed `Hadamard index` (in binary format).

```
binHadIdx = fliplr(dec2bin(HadIdx,M));    % Bit reversing of the binary index
binHadIdx = uint8(binHadIdx)-uint8('0'); % Convert from char to integer array
binSeqIdx = zeros(N,M-1,'uint8');        % Pre-allocate memory
for k = M:-1:2
    % Binary sequency index
    binSeqIdx(:,k) = xor(binHadIdx(:,k),binHadIdx(:,k-1));
end
SeqIdx = bin2dec(int2str(binSeqIdx));    % Binary to integer sequency index
walshMatrix = hadamardMatrix(SeqIdx+1,:) % 1-based indexing
walshMatrix =

     1     1     1     1     1     1     1     1
     1     1     1     1    -1    -1    -1    -1
     1     1    -1    -1    -1    -1     1     1
     1     1    -1    -1     1     1    -1    -1
     1    -1    -1     1     1    -1    -1     1
     1    -1    -1     1    -1     1     1    -1
     1    -1     1    -1    -1     1    -1     1
     1    -1     1    -1     1    -1     1    -1
```



fft
ifft
ifwht(eye(N))

CDMA via Hadamard Matrix

```
N = 8;                                   % 8 Users
H = hadamard(N);                         % Hadamard matrix
%% At transmitter(s),
S = [8    0    12    0    18    0    0    10];
r = S*H
% r = 8.*H(1,:) + 12.*H(3,:) + 18.*H(5,:) + 10.*H(8,:);
% Alternatively, use
% r = ifwht(S,N,'hadamard')
%% At Receiver,
S_hat = (1/N)*r*H'
% Alternatively, use
% S_hat = fwht(r,N,'hadamard')
```

$s(1)H(1,:) + s(2)H(2,:) + \cdots$

$$\hat{s}_k = \frac{1}{N}\langle \underline{r}, \underline{c}_k \rangle = \frac{1}{N}\underline{r} * \underline{c}_k^H$$

$$[\hat{s}_1 \ \hat{s}_2 \ \cdots \ \hat{s}_N] = \hat{\underline{s}}$$

Discrete Walsh-Hadamard transform

Specify the order of the Walsh-Hadamard transform coefficients. ORDERING can be 'sequency', 'hadamard' or 'dyadic'. Default ORDERING type is 'sequency'.