Elektronik Labor

OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG
ELEKTRO- UND INFORMATIONSTECHNIK

WSAN

Wireless Sensor / Actuator Networks

Auxiliary Pages

Prof. Dr. Martin J. W. Schubert, Electronics Laboratory

WSAN Auxiliary Pages

Abstract. This scipt offers some auxiliary pages to scipt WSAN.

1 Introduction

The WSAN script was initially created by MAP master student Albert Martinez in 2014-2015 and extended and improved by MEM master student Matthias Wagner in 2015-2016.

The pages below offer some additional information by Prof. M. Schubert.

The organization of this document is as follows:

~~Chapter 1 - This introduction,~~

~~Chapter 2 - Fields and Waves Theory,~~

~~Chapter 3 - Offers additional information about the MSP430 hardware~~

~~Chapter 4 - Details some data transmission principles~~

~~Chapter 5 - Explains required C software features~~

~~Chapter 6 - Gives some references~~

2 Fields and Waves

2.1 RF Circuit Design

Radio frequency (RF) circuit design has to respect wavelength. Classical circuit simulators have „nodes“, RF circuits have wires. The voltage versus time and space is discribed by the wave equation

$$U(\vec{r},t) = U\sin(\omega t - \vec{k}\cdot\vec{r})$$

Period: $T = \frac{2\pi}{\omega}$,

Wavelength: $\lambda = \frac{2\pi}{|\vec{k}|}$

2.2 Elektro-Magnetic Waves in Free Space

$c_0 = \frac{1}{\sqrt{\varepsilon_0\mu_0}} = 2{,}997\,924\,58\;10^8$ m/s $\cong$ $3\;10^8$ m/s, speed of light in vacuum

$c = \frac{1}{\sqrt{\varepsilon\mu}} = \frac{c_0}{\sqrt{\varepsilon_r\mu_r}}\left[\frac{m}{s}\right]$, speed of light in materials and $c = \lambda\cdot f$

$Z_0 = \sqrt{\frac{\mu_0}{\varepsilon_0}} = 376{,}730\,396\,2\;\Omega \cong 376{,}730\,396\,2\;\Omega$ wave impedance in vaccum

$Z = \sqrt{\frac{\mu}{\varepsilon}} = Z_0\sqrt{\frac{\mu_r}{\varepsilon_r}} \cong 377\Omega\sqrt{\frac{\mu_r}{\varepsilon_r}}$ [Ω], wave impedance in materials

$\vec{S} = \vec{E}\times\vec{H}\left[\frac{VA}{m^2}\right]$, Pointing Vektor, intesitiy and direction of power density flow

Table 2.2-1: Permeability numbers μ_r for $\mu = \mu_r\,\mu_0$ in H/m=Vs/Am

diamagnetisch: $\mu_r<1$	Vakuum	ferromagnetisch
Kupfer: μ_r=0,999 990 4	μ_r=1	Eisen: μ_r=50 ... 500
Wasser: μ_r=0,999 990 97	$\mu_0 = 1{,}256\,637\,061\,44\;10^{-6}$ H/m	Baustahl: μ_r=800 ... 2000

Table 2.2-2: Dielectric number ε_r for $\varepsilon = \varepsilon_r\,\varepsilon_0$ in F/m=As/Vm

$\varepsilon_r<1$	Vakuum	$\varepsilon_r>1$
	ε_r=1	Glas: ε_r=3...15
	$\varepsilon_0 = 8{,}854\,187\,82\;10^{-12}$ F/m	Gummi: ε_r=2,5...3,5
		Wasser: ε_r=81

3 Hardware: MSP430

3.1 Behavioral GPIO Port-Model

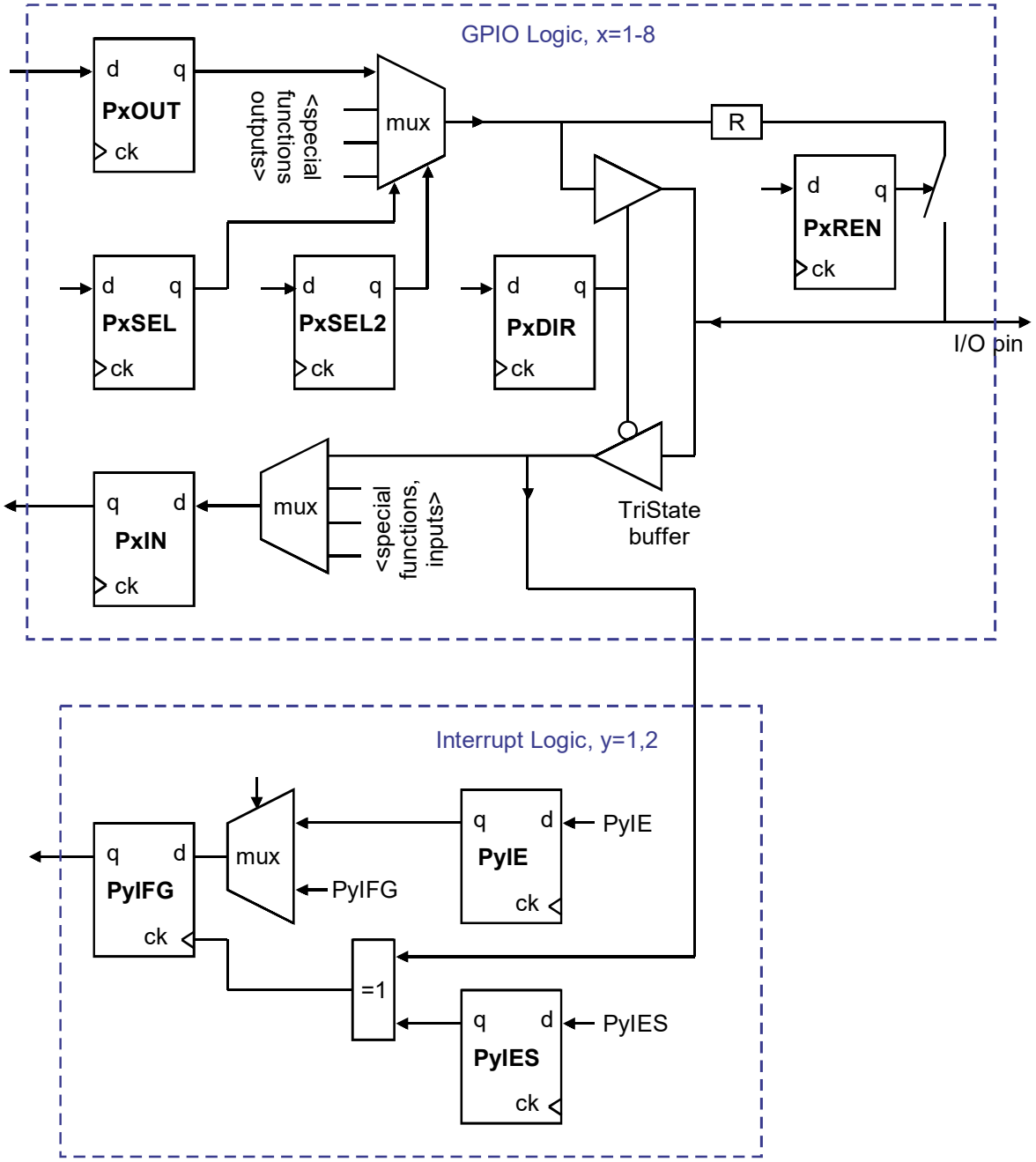


Fig. 3.1: General Purpose I/O (GPIO) Port simplified behavioral model of MSP430

4 Wireless Transmission

4.1 Channel Spacing for MSP430

Exact computations:

Defaults: f_{xosc}=26MHz, FREQ=0x22B13B=2273595,
CHAN=0x14=20, CHANSP_M=0xF8=248, CHANSPC_E=0b10=2.

With f_{xosc}=26MHz we get

$$f_{carrier} = f_{xosc}\, 2^{-16}\, (FREQ + CHAN\, ((256 + CHANSPC\_M)\, 2^{CHANSPC\_E-2})$$
$$\sim 396.729Hz\, (FREQ + CHAN\, ((256 + CHANSPC\_M)\, 2^{CHANSPC\_E-2})$$

Let CHANNR=0 to obtain the minimum offset carrier frequency with 0≤FREQ≤0xFFFFFF set

$$f_{coff} = f_{xosc}\, 2^{-16}\, FREQ$$
$$f_{coffmin} = f_{xosc}\, 2^{-16}\, FREQ = 26MHz\, 2^{-16} = 0.000\ MHz$$
$$f_{coffdef} = f_{xosc}\, 2^{-16}\, FREQ = 26MHz\, 2^{-16}\, 0x22B13B = 902.000\ MHz$$
$$f_{coffmax} = f_{xosc}\, 2^{-16}\, FREQ = 26MHz\, 2^{-16}\, 0xFFFFFF = 6656.000\ MHz$$

Total carrier Frequency is consequently:

$$f_c = f_{coff} + CHAN\ \Delta f_c$$

with cannel spacing Δf_c computed from

$$\Delta f_c = f_{xosc}\, 2^{-16}\ ((256 + CHANSPC\_M)\, 2^{CHANSPC\_E-2})$$
$$\sim 396.73Hz\, ((256 + CHANSPC\_M)\, 2^{CHANSPC\_E-2}$$

The minimum, default, maximum channel spacing is

$$\Delta f_{cmin} = f_{xosc}\, 2^{-16}\, ((256 + CHANSPC\_M)\, 2^{CHANSPC\_E\,-\,2}\,)$$
$$= 26MHz\, 2^{-16}\, (256 + 0)\, 2^{0-2} \sim 25.391\ KHz$$

$$\Delta f_{cdef} = f_{xosc}\, 2^{-16}\ (256 + CHANSPC\_M)\, 2^{CHANSPC\_E\,-\,2}$$
$$= 26MHz\, 2^{-16}\, (256 + 248)\, 2^{2-2} = 199.951kHz \sim 200\ KHz$$

$$\Delta f_{cmax} = f_{xosc}\, 2^{-16}\, ((256 + 255)\, 2^{CHANSPC\_E\,-\,2}\,)$$
$$= 26MHz\, 2^{-16}\, (256 + 255)\, 2^{3-2} \sim 404.663\ KHz$$

Knowing CHAN is an 8-bit integer delivers:

$$f_{cdef} = f_{cdefoff} + CHAN\ \ \Delta f_{cdef}$$
$$f_{cdefmin} = f_{cdefoff} + 0 = 902.000\ MHz$$
$$f_{cdefmin+1} = f_{cdefoff} + 1\ 199951Hz = 902.200\ MHz$$
$$f_{cdefdef} = f_{cdefoff} + 20\ 199951Hz = 905.999\ Mhz$$
$$f_{cdefdef} = f_{cdefoff} + 255\ 199951Hz = 952.988\ MHz$$

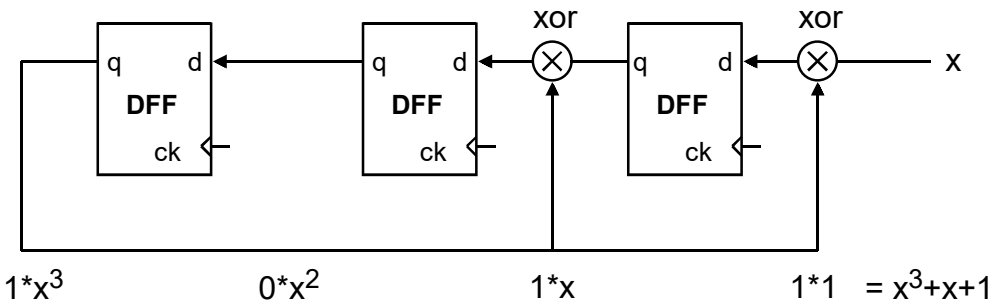
Exercise:

f_{coff} 868MHz, ChannelSpacing Δf_c=200KHz, CHAN=0x14. Compute f_c!

Solution: $f_c = f_{coff} + CHAN\ \Delta f_c = 868MHz + 20\ 0.2MHz = 868MHz + 4MHz = 872MHz$

4.2 CRC: Cyclic Redundancy Check

Initial 1's avoid that leading 0's are overlooked.
Append trailing 0's
Define generator polynomial
Compute CRC checksum
Transmitter: Send data and checksum
Receiver: verify checksum is all 0's



Legend:

data
generator polynomial
initial 1's
trailing 0's
CRC check sum at sender
000 valid receiver checksum

Sender:

```
111101101000
1011
01000
 1011
 001111
   1011
   01000
    1011
    001110
      1011
      01010
       1011
       00010
```

Receiver:

```
111101101010
1011
01000
 1011
 001111
   1011
   01000
    1011
    001110
      1011
      01011
       1011
       00000
```

5 Software: C

5.1 Basic Operations

In C a variable X is assumed to be **`false`** if `X==0` and **`true`** otherwise.
Locig operators can operate on total variable (e.g. &&, ||, ^^, !) or bitwise (e.g. &, |, ^, ~)

Let
```
int8_t BIT1 = 0b00000010, X;

X = X + 1; // increment X
X += 1;    // increment X, short form of line above
X = !X; // NOT operation on total variable, result is 0 or 1.
X = ~X; // bitwise NOT operates on bits of variable X
```

5.1.1 Set a particular Bit to '1':

C uses OR for total variable || and bitwise OR |
Principle: Assume 1-bit variable a. Then '1' | a → '1', '0' | a → a.
```
X = X | BIT1;
X |= BIT1;   // short form of line above
X += BIT1;   // dangerous! Works only if considered bit = '0' before
```

5.1.2 Set a particular Bit to '0':

C uses AND for total variable && and bitwise AND &
Principle: Assume 1-bit variable a. Then '1' & a → a, '0' & a → '0'.

```
int8_t notBIT1;
notBIT1 = ~ BIT1; // == 0b11111101
X = X & notBIT1;  // clear bit 1 (i.e. set it to '0')
X = X & (~BIT1;)  // same effect as line above
X &= ~BIT1;       // same effect as 2 lines above
```

5.1.3 Toggle a particular Bit:

C uses EXOR (exclusive OR) for total variable ^^ and bitwise EXOR ^

Principle:

Truth table for a EXOR b: y = a ^ b

b	a	y	delivers:
0	0	0	
0	1	1	a unchanged if b=0
1	0	1	
1	1	0	a inverted if b=1

```
int8_t notBIT1;
X = X ^ BIT1;  // toggles BIT1 only
X ^= BIT1;     // same effect as line above
```

5.2 Data types

5.2.1 typedef

typedef introduces a new type name for a data type. Variables are declarated when using it.

Wrong:
```
typedef double real_t;
real_t = 1.7; // wrong: assigning value to a type
```

Right:
```
typedef double real_t;
real_t dblvar;
dblvar = 1.7;
```

5.2.2 enum

Right:
```
// enum values: black=0, blue=1, cyan=2,...
enum color_t {black,  blue,    green,   cyan,
              red,    magenta,  brown,  lightGray };
color_t color;
color = blue;
```

5.2.3 Pointer

```
int i1=0x1a1b1c1d, i2;
int *pA=&i1;
i2 = 0x1a1b1c1d;
pA = &i2;
pA = &pA;
```

Name	Typ	&Name
i1	**int**	**0 x 0 2 2 f f 6 4**

0 x 1 a 1 b 1 c 1 d	Wert
0 x 0 2 2 f f 6 4	Adr.

Name	Typ	&Name
i2	**int**	**0 x 0 2 2 f f 6 0**

0 x 2 a 2 b 2 c 2 d	Wert
	Adr.

pA=&i1:

Name	Typ	&Name
pA	**int ***	**0 x 0 2 2 f f 5 c**

	Wert
	Adr.

pA=&i2:

Name	Typ	&Name
pA	**int ***	**0 x 0 2 2 f f 5 c**

	Wert
	Adr.

pA=&pA:

Name	Typ	&Name
pA	**int ***	**0 x 0 2 2 f f 5 c**

	Wert
	Adr.

Fig. 3.1.3(a): Pointer exercise

Name	Typ	&Name
i1	int	0 x 0 2 2 f f 6 4

0 x 1 a 1 b 1 c 1 d	Wert
0 x 0 2 2 f f 6 4	Adr.

Name	Typ	&Name
i2	int	0 x 0 2 2 f f 6 0

0 x 2 a 2 b 2 c 2 d	Wert
0 x 0 2 2 f f 6 0	Adr.

pA=&i1:

Name	Typ	&Name
pA	int *	0 x 0 2 2 f f 5 c

0 x 0 2 2 f f 6 4	Wert
0 x 0 2 2 f f 5 c	Adr.

pA=&i2:

Name	Typ	&Name
pA	int *	0 x 0 2 2 f f 5 c

0 x 0 2 2 f f 6 0	Wert
0 x 0 2 2 f f 5 c	Adr.

pA=&pA:

Name	Typ	&Name
pA	int *	0 x 0 2 2 f f 5 c

0 x 0 2 2 f f 5 c	Wert
0 x 0 2 2 f f 5 c	Adr.

Fig. 3.1.3(b): Solution to pointer exercise

5.2.4 Functions

```
int x=5, y;

int square(int a){ return a*a) };
void square_by_ref(int *aref) { *aref = *aref * (*aref); }

y = square(x);      // call by value
square_by_ref(&x);  // call by reference
```

Note: Call by reference allows to write on argument!

5.2.5 Callback Functions

```
int square(int a)
{ return a*a);}

int fCallBack(int (*f)(int), int x)
{ return f(x); }

void main(void)
{ int x=5,y;
  y = fCallBack(&square,x);
}
```

6 Interface Board Launchpad – Sensor Node

An interface board of Stefan Zenger [8] between between Launchpad MSP43FR6989 and target board named “1-sensor-node” allows to program the CC430 on the target board.

6.1 Installation

- Remove all jumpers from the Launchpad
- Plug in the interface board such that most (nearly all) Launchpad pins are plugged
- Plug 1-sensor-node on top of the interface board, watch out for “µC this side” mark

6.2 Operating the LEDs / UART

6.2.1 Operating the LEDs

On top of the interface board there are 6 diodes. 2 of them can be addressed by the sensor-node board, the other 4 are controlled of the MSP430 µC.

Controlling µC	LED	PIN
CC430	LED_RX	P1.5/UCA0RXD
CC430	LED_RX	P1.6/UCA0TXD
MSP430	0	P1.7
MSP430	1	P1.6
MSP430	2	P5.3
MSP430	3	P1.3

6.2.2 UART

Both pins connected to LED_RX/TX can be used for UART connections. Set jumpers J_RX, J_TX.

6.3 Example Code

Listing 6.3 should cause blinking diodes LED_RX, LED_TX.

Listing 6.3: example code for interface board

```
#include <msp430.h>


/**
 * main.c
 */
int main(void)
{
    WDTCTL = WDTPW + WDTHOLD;                 // Stop WDT
    P1DIR |= BIT5 + BIT6;                           // P1.0 output
    //P1DIR = 0;
    P1OUT = 0;

    __no_operation();

    P1OUT = BIT6;

    __no_operation();

    TA1CCTL0 = CCIE;                          // CCR0 interrupt enabled
    TA1CCR0 = 50000;
    TA1CTL = TASSEL_2 + MC_2 + TACLR;         // SMCLK, contmode, clear TAR

    __bis_SR_register(LPM0_bits  +  GIE);           //  Enter  LPM0,  enable
interrupts
    __no_operation();                         // For debugger
   }

    // Timer A0 interrupt service routine
    #if defined(__TI_COMPILER_VERSION__) || defined(__IAR_SYSTEMS_ICC__)
    #pragma vector=TIMER1_A0_VECTOR
    __interrupt void TIMER1_A0_ISR(void)
    #elif defined(__GNUC__)
    void __attribute__ ((interrupt(TIMER1_A0_VECTOR))) TIMER1_A0_ISR (void)
    #else
    #error Compiler not supported!
    #endif
    {
      P1OUT ^= BIT5;                            // Toggle P1.0
      P1OUT ^= BIT6;                            // Toggle P1.0
      //TA1CCR0 += 50000;                         // Add Offset to CCR0
    }
}
```

7 Transmission

7.1 Open Systems Interconnect (OSI) Model

Table 7.1: Layers according to the OSI Model

#	Layer Name	Data Unit	Protocol	Address
7	Application		Telnet, SSH, SSL, SCP	hostname
6	Presentation		E-mail	user@domain
5	Session		Web browser	http://www...
4	Transport	Segment	TCP (Transmission Control Protocol)	Port Number
3	Network	Packet	IP (Internet Protocol)	IP address
2	Data Link	Frame	Interface Device (example: MRFI)	MAC address
1	Physical	Bit	Ethernet, WiFi,… (example: CC430)	

1. Physical Layer: [https://en.wikipedia.org/wiki/OSI_model]

The physical layer defines the electrical and physical specifications of the data connection. It defines the relationship between a device and a physical transmission medium (for example, an electrical cable, an optical fiber cable, or a radio frequency link). This includes the layout of pins, voltages, line impedance, cable specifications, signal timing and similar characteristics for connected devices and frequency (5 GHz or 2.4 GHz etc.) for wireless devices. It is responsible for transmission and reception of unstructured raw data in a physical medium. Bit rate control is done at the physical layer. It may define transmission mode as simplex, half duplex, and full duplex. It defines the network topology as bus, mesh, or ring being some of the most common.

The physical layer is the layer of low-level networking equipment, such as some hubs, cabling, and repeaters. The physical layer is never concerned with protocols or other such higher-layer items. Examples of hardware in this layer are network adapters, repeaters, network hubs, modems, and fiber media converters.

Example: Wireless connection with CC430 devices

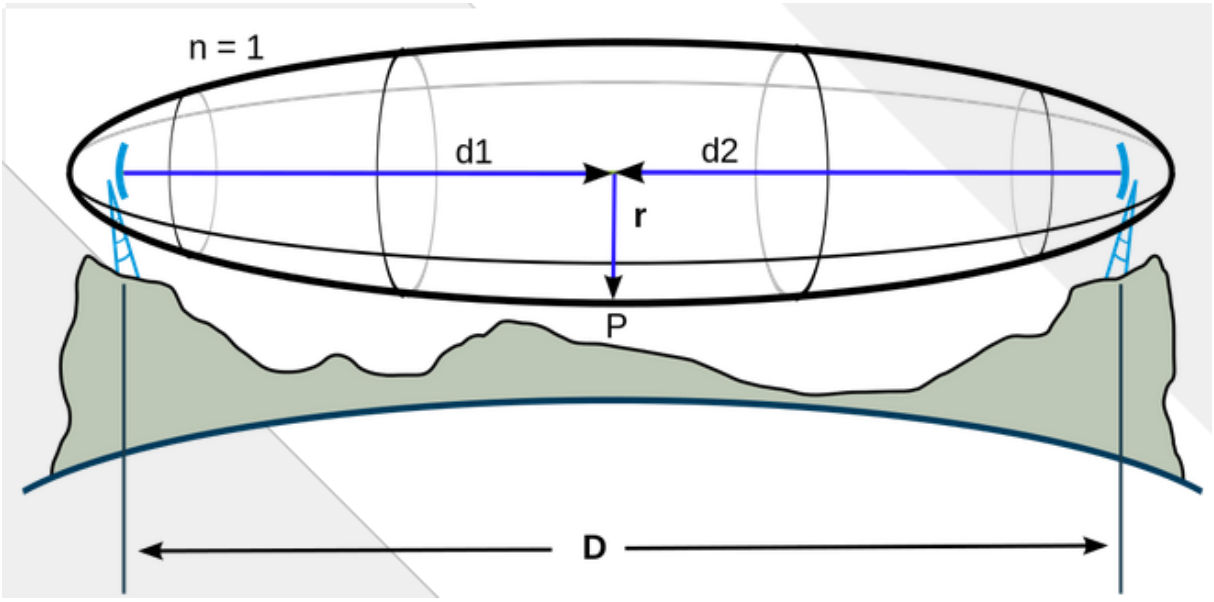
1. Data Link Layer: [https://en.wikipedia.org/wiki/OSI_model]

The data link layer provides node-to-node data transfer—a link between two directly connected nodes. It detects and possibly corrects errors that may occur in the physical layer. It defines the protocol to establish and terminate a connection between two physically connected devices. It also defines the protocol for flow control between them.

Example: MRFI

7.2 Fresnel Zone

The Fresnel zone is the space that must be free of barriers for undisturbed radio transmission, it has she shape of a Zeppelin.



8 RF Modulationstechniken

Quadratur-Amplituden-Modulation (QAM) ist heutzutage die dominierende Modulationstechnik im RF-Bereich. Im Folgenden werden kurz deren Prizipien erläutert.

8.1 Amplitudenmodulation (AM, ASK)

8.1.1 Analog: Frequenzmodulation

Die Änderung der Amplitude transportiert das Signal s(t), die Trägerfrequnez bleibt konstant.

$U_{AM} = U_0 \cdot \sin(\omega_0 t) \cdot (1-m) \cdot s(t)$

M= Modulationsgrad, es muss 0<m<1 sein, sonst entstehen die in den unteren Bildteilen dargestellten Übersteuerungsprobleme.

Modulation: Mischer

Demodulation:
Früher: Diode + RC-Tiefpass
Heute: PLL + Downsampling

Anwendungsbeispiele:
AM-Rundfunk (LW, MW, KW)

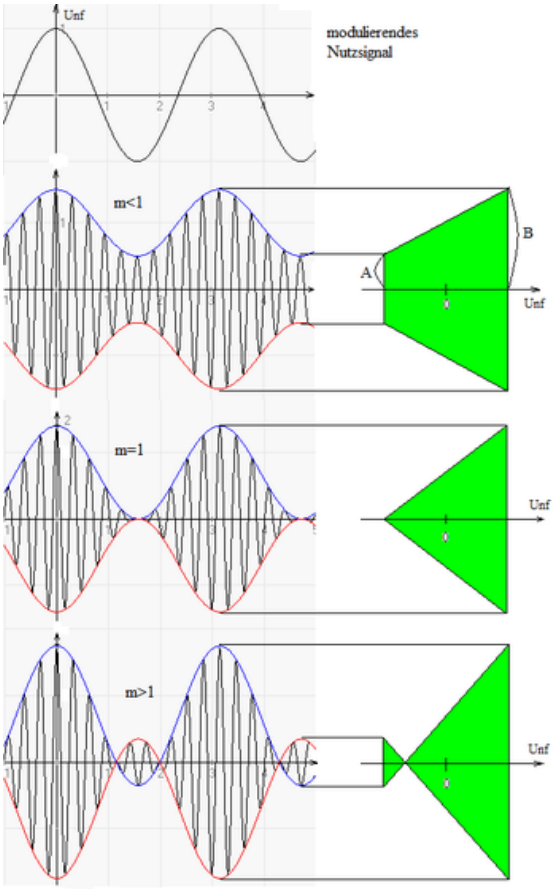


Bild aus Wikipedia[9]: "Amplitudenmodulation"

8.1.2 Digital: ASK (Amplitude Shift Keying)

Senden eines sinusförmigen Trägersignals der Amplitude $\Delta U \cdot 2^N - U_{off}$. Durch entsprechendes setzen von U_{off} kann der Träger mit einer positiven oder negativen Amplitude moduliert werden.

N=1: 2-ASK oder Bniary ASK, u.a. auch Morsen. (2-ASK ist identisch mit 2-PSK)
N=2: 4-ASK
N=3: 8-ASK
....

8.2 Frequenzmodulation (FM, FSK)

8.2.1 Analog: Frequenzmodulation

Die Änderung der Frequenz transportiert das Signal s(t), die Amplitude bleibt konstant.

$U_{FM} = U_0 \cdot \sin[(\omega_0 + \Delta\omega \cdot s(t)) \cdot t]$

Modulation: meist mit VCO (Voltage Controlled Oscillator)

Demodulation: mit PLL (Phase Locked Loop, VCO in der Rückkopplungsschleife)

Anwendungen:
Audio/Video-Technik,
Messtechnik
FM-Rundfunk (UKW)
Einige Video-Band-Recorder
Die Fernsehnorm SECAM überträgt die Farbinformation mit FM.

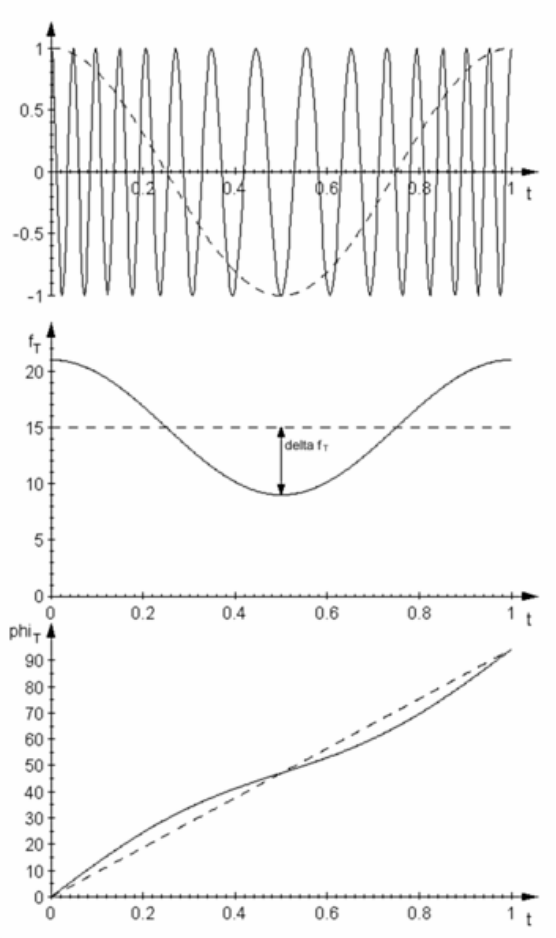


Bild aus Wikipedia [9]: "Frequenzmodulation"

8.2.2 Digital: Frequnzumtastung bzw. FSK (Frequency Shift Keying)

Prinzip: Eine Frequenz ist vorhanden oder nicht. Bei 8 verschiedenen Frequenzen kann jeweils 1 Byte gleichzeitig übertragen werden. Die Frequenzen müssen so verschachtelt werden, dass die hohen Frequenzen nicht auf den Oberwellen der tieferen Frequenzen liegen, da diese schon durch einfache Verformung (z.B. Übersteuerung) der Grundwelle entstehen.

Anwendung: z.B. FAX

8.3 Phasenmodulation (PM, PSK)

8.3.1 Analog: Phasenmodulation

Die Änderung der Phase transportiert das Signal s(t), die Amplitude bleibt konstant. Im Ergebnis ist PM optisch der FM sehr ähnlich.

$U_{PM} = U_0 \cdot \sin[\omega_0 t + \Delta\varphi \cdot s(t)]$

8.3.2 Digital: Phasenumschaltung oder PSK (Phase Shift Keying)

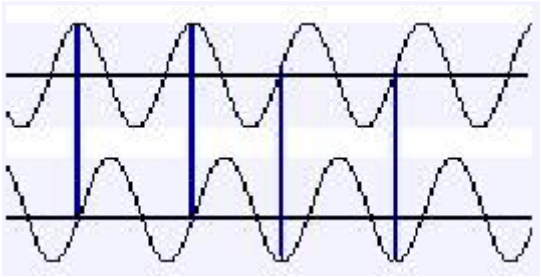
Anwendungen:
Mobiltelefone (GSM), Bluetooth, WLAN....

2-PSK oder binäre PSK: Im Ergebnis gleich zur AM mit Amplitudenumschaltung ±1.
4-PSK oder QPSK oder Quadruple PSK: Die Phase ändert sich um M·90°, M=0,1,2,3. 4.PSK ist im Ergebnis identisch mit bestimmten Realisierungen der QAM (siehe unten)
8-PSK: Die Phase ändert sich um M·45°, M=0,1,2,3...7

PSK-Sonderformen (Text kopiert aus Wikipedia [9], DPSK):
DPSK: *Differential Phase Shift Keying* (DPSK) Nur der Wechsel von einer logischen „1" zu einer logischen „0" oder umgekehrt bewirkt bei der Modulation einen Phasensprung. Findet kein Phasensprung statt, bleibt die letzte Information gültig.
SDPSK: *Symmetrical Differential Phase Shift Keying* (SDPSK) ist die Phasenverschiebung symmetrisch. Eine positive Phase von 90° entspricht dem Bit 1, eine negative Phase von 90° dem Bit 0

8.4 Quadratur-Amplitudenmodulation (QAM)

Wo sin(x)=0 ist, hat cos(x) extrema und umgekehrt. Somit ist I(t)·cos(t)=±I(t) wenn Q(t)·sin(t)=0 und umgekehrt. Auf diese Punkte konzentriert sich die Abtastung.
(Bild rechts kopiert von http://de.wikipedia.org/wiki/Bild:Quadraturmodulation_sinus_cosinus.png#file)



Entscheidendes Kriterium für die digitale QAM ist die BER: Bit Error Rate

Quadrature amplitude modulation (Text taken from Wikipedia, engl. Version, [9])

8.4.1 Analog QAM

$$s(t) = I(t)\cos(2\pi f_0 t) + Q(t)\sin(2\pi f_0 t)$$

,

where $I(t)$ and $Q(t)$ are the modulating signals and f_0 is the carrier frequency.

In the ideal case $I(t)$ is demodulated by multiplying the transmitted signal with a cosine signal:

$$\begin{aligned} r_i(t) &= s(t)\cos(2\pi f_0 t) \\ &= I(t)\cos(2\pi f_0 t)\cos(2\pi f_0 t) + Q(\end{aligned}$$

Using standard trigonometric identities, we can write it as:

$$\begin{aligned} r_i(t) &= \frac{1}{2}I(t)\left[1+\cos(4\pi f_0 t)\right] + \frac{1}{2}Q(t)\sin(4\pi f_0 t) \\ &= \frac{1}{2}I(t) + \frac{1}{2}\left[I(t)\cos(4\pi f_0 t) + Q(t)\sin(4\pi f_0 t)\right] \end{aligned}$$

Topics in Modulation techniques

Analog modulation

AM | SSB | FM | PM | **QAM**

Digital modulation

OOK | ASK | PSK | FSK | MSK | **QAM** | CPM | TCM | OFDM

Spread spectrum

FHSS | DSSS

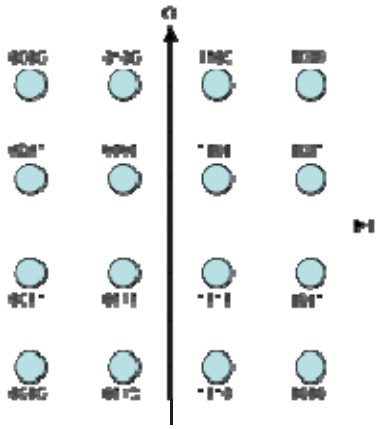
edit

8.4.2 Digital QAM

Like for many digital modulation schemes, the constellation diagram is a useful representation which is relied upon in this article.

8.4.2.1 Rectangular QAM

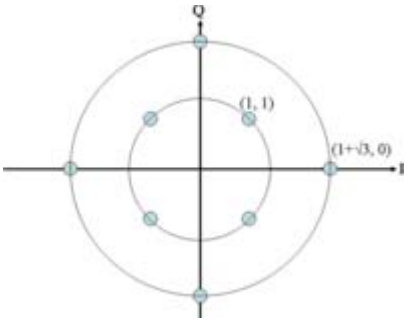
Rectangular QAM constellations are, in general, sub-optimal in the sense that they do not maximally space the constellation points for a given energy. However, they have the considerable advantage that they may be easily transmitted as two pulse amplitude modulation (PAM) signals on quadrature carriers, and can be easily demodulated. The non-square constellations, dealt with below, achieve marginally better bit-error rate (BER) but are harder to modulate and demodulate.



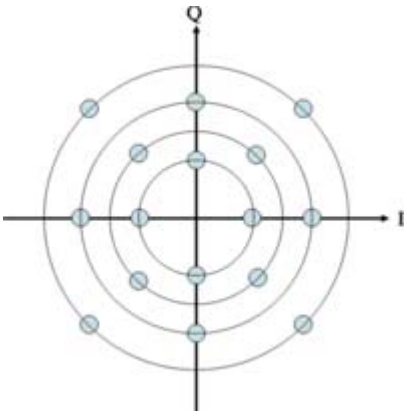
Constellation diagram for rectangular 16-QAM.

The first rectangular QAM constellation usually encountered is 16-QAM, the constellation diagram for which is shown here. A Gray coded bit-assignment is also given. The reason that 16-QAM is usually the first is that a brief consideration reveals that 2-QAM and 4-QAM are in fact binary phase-shift keying (BPSK) and quadrature phase-shift keying (QPSK), respectively. Also, the error-rate performance of 8-QAM is close to that of 16-QAM (only about 0.5dB better), but its data rate is only three-quarters that of 16-QAM.

8.4.2.2 Non-rectangular QAM (interested students only)



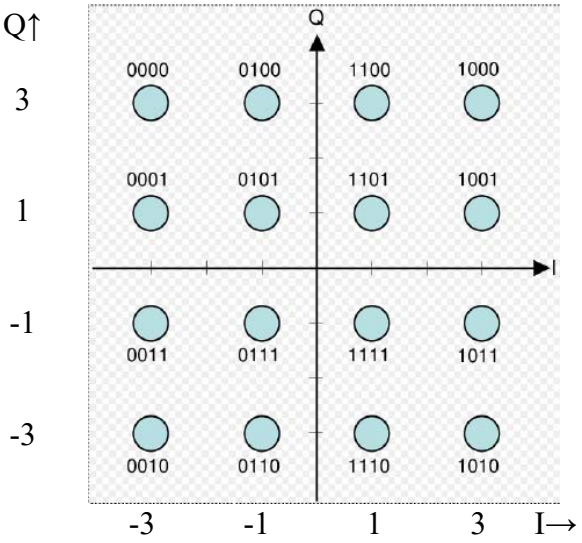
Constellation diagram for circular 8-QAM.



Constellation diagram for circular 16-QAM.

It is the nature of QAM that most orders of constellations can be constructed in many different ways and it is neither possible nor instructive to cover them all here. This article instead presents two, lower-order constellations.

8.4.2.3 Exercise for QAM-Modulation



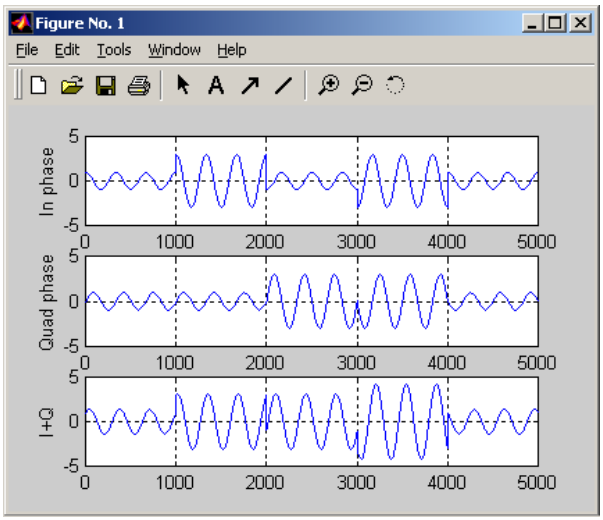


Bild **(a)** QAM16-Modulation-Code **(b)** QAM modulatted time-domain signal

Zeitachse	Amplitude I	Amplitude Q	Coded bin value	Coded hex value
0000...1000	1	1	1101	D
1000...2000				
2000...3000				
3000...4000				
4000...5000				

Solution:

Zeitachse	Amplitude I	Amplitude Q	Coded bin vlaue	Coded hex vlaue
0000...1000	1	1	1101	D
1000...2000	3	1	1001	9
2000...3000	-1	3	0100	4
3000...4000	-3	-3	0010	2
4000...5000	1	-1	1111	F

9 RF SoC: The Superheterodyne Transceiver

Fig. 1 from [11]: Basic structure of RF transmitters. This example illustrates interferences between mobile phone (GSM) and WLAN.

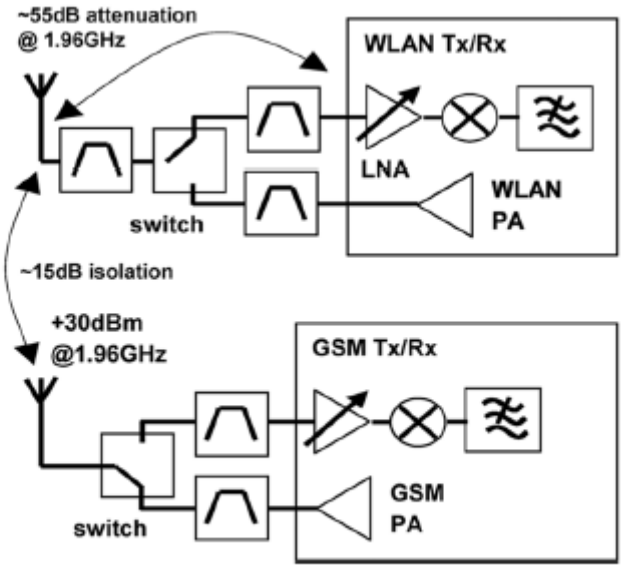


Fig. 1. Interferer situation for cellular/802.11 coexistence issues.

Transceiver is a synonym for a combination of a transmitter (TX) and a receiver (RX).

9.1 I/Q-Receiver

Fig. 1 of [5]:

$$f_s = \frac{4f_c}{2K-1}, \quad K = 1, 2, 3, \ldots$$

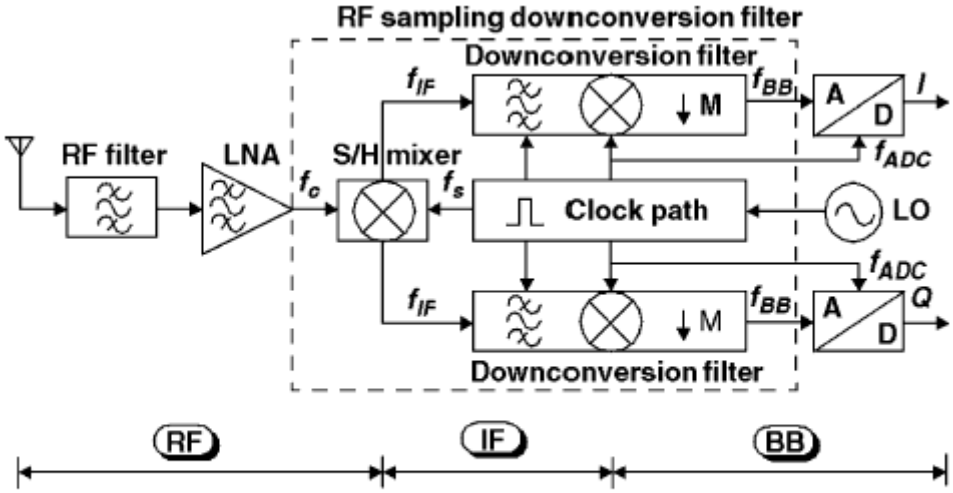


Fig. 1. RF sampling receiver front-end architecture with the corresponding frequency partitioning.

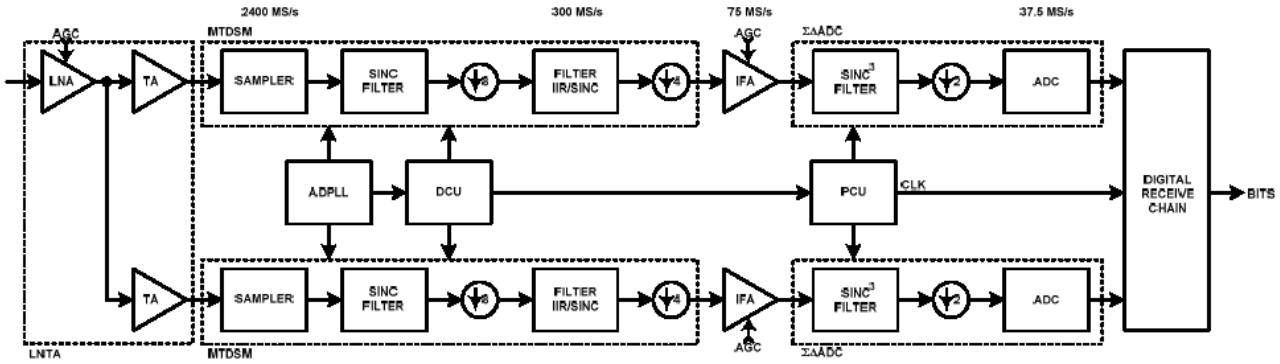


Fig 15.1.1 from [13]: Receiver Block Diagram

BB	Baseband
IF	Intermediate Frquency
LAN	Local Area Network
LNA	Low Noise Amplifier
PA	Power Amlifier
PGA	Programmable Gain Amplifier
RF	Radio Frequency
RSSI	Received Signal Strength Indicator
WLAN	Wireless LAN

10 References

[1] G. Kupris, Seminar “Drahtlose Sensornetzwerke”, Regensburg, 08.03.2017

[2] G. Kupris, A. Sikora: ZigBee - Datenfunk mit IEEE 802.15.4 und ZigBee, Franzis Verlag München 2007

[3] R. Gessler, T. Krause: Wireless-Netzwerke für den Nahbereich, 2. Auflage, Springer Vieweg Wiesbaden 2015

[4] M. Kraußе, R. Konrad: Drahtlose ZigBee-Netzwerke - Ein Kompendium, Springer Vieweg Wiesbaden 2014

[5] R. Faludi: Building Wireless Sensor Networks, O‘Reilly 2012

[6] J. Titus: The Hands-On Xbee Lab Manual, Elsevier Publishing 2012

[7] J. Wright, J. Cache: Hacking Exposed Wireless, Mc Graw Hill Education 2015

[8] S. Zenger, *MSP-EXP430FR4133 Apadter for 1-Sensor-Node_V4*, OTH Regensburg 20. Feb. 2018.

[9] http://www.wikipedia.org

[10] K. Muhammad, T. Murphy, R. B. Staszewski, "Verification of RF SoCs: RF, Analog, Baseband and Software", IEEE

[11] O. Charlton et al. "A Low-Power High performance SiGe BiCMOS 802.11a/b/g Transceiver IC for Cellular and Bluetooth Co-Existence Applications", IEEE Journal of Solid-State Ciruits (SSC), Vol 41, No. 7, July 2006, pp.1503-1512.

[12] D. Jakonis et al. "A 2.4-GHz RF Sampling Receiver Front-End in 0.18-µm CMOS", IEEE Journal of Solid-State Ciruits (SSC), Vol 40, No. 6, Juny 2006, pp.1265-1277.

[13] K. Muhammad et al., "A Discrete Time Bluetooth receiver in a 0.13µm Digital CMOS Process", ISSCC 2004 / Session 15 / Wireless Consumer ICs.