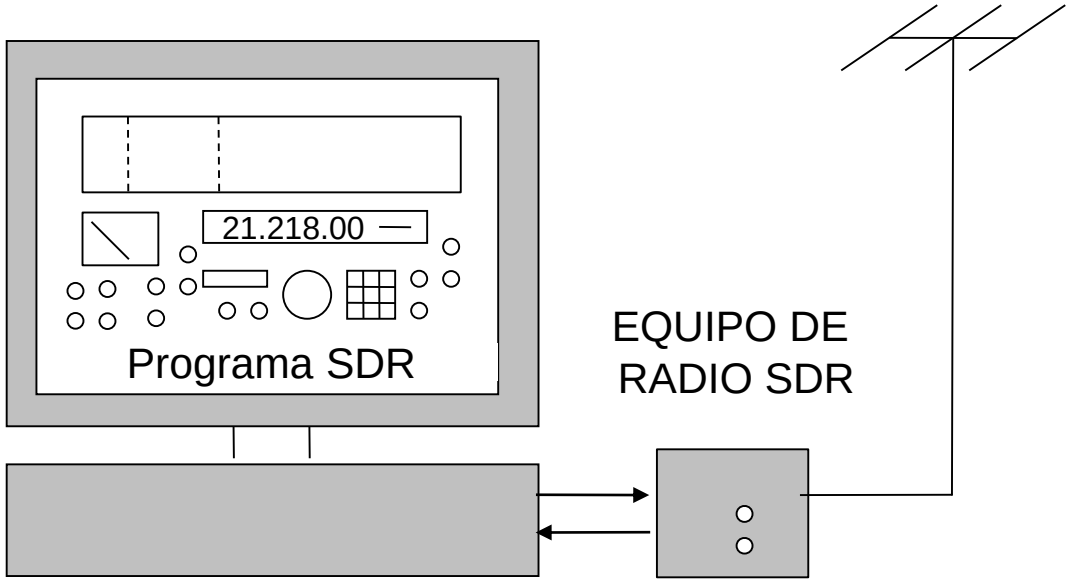
SDR: EQUIPOS DE RADIO DEFINIDOS POR *SOFTWARE*

Albacete, 5 de diciembre de 2010

Sergio Manrique, EA3DU

EQUIPO DE RADIO
CONVENCIONAL

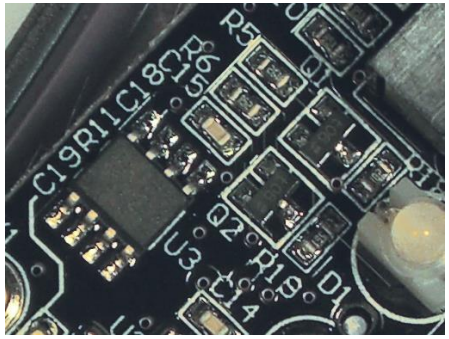


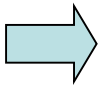
Una definición de SDR

- Equipo cuyas características de modulación y demodulación están definidas únicamente en forma de *software*
- Su *hardware* es lo bastante flexible para soportar modulaciones de distintos anchos de banda y formatos, y cambiar de unas a otras rápidamente
- Posee la capacidad de incorporar nuevo *software*, añadiendo funcionalidades o mejorando las existentes
- Incluye un interfaz para controladores externos
- Origen: años 70 (Dpto. Defensa EEUU)
- Primera definición del concepto SDR: 1991

Características de los equipos SDR

- Sustituyen la implementación de funciones en forma de “hardware” a favor de “software”, sea *software* interno o esté en un ordenador externo. Ejemplo: procesado de señal (filtrado, modulación/demodulación, análisis espectral, etc.)





```
case AMdet:
      for (i = 0; i < am->size; i++)
                  {
                    am->lock.curr = Cmag (CXBdata (am->ibuf, i));
                    am->dc = 0.9999f * am->dc + 0.0001f * am->lock.curr;
                    am->smooth = 0.5f * am->smooth + 0.5f * (am->lock.curr
- am->dc);
                    /* demout = am->smooth; */
                    CXBdata (am->obuf, i) = Cmplx (am->smooth, am-
>smooth);
                  }
```

- Posibilidades de experimentación

- Permiten cambios en su funcionalidad sin tener que adquirir dispositivos adicionales

- Se reduce la distorsión generada por amplificadores y mezcladores: mayor linealidad y mejor control de los AGC

Características de los equipos SDR

- Mejora continua, mediante actualizaciones del software
- El DSP permite corregir imperfecciones debidas al canal de RF: ecualización, etc.
- Menor consumo de energía

```
/* am_demod.c */


#include <am_demod.h>
#include <cxops.h>

/*---------------------------------------------------------------------------*/
/* private to AM */
/*---------------------------------------------------------------------------*/

static void
init_pll (AMD am,
                        REAL samprate, REAL freq, REAL lofreq, REAL hifreq, REAL bandwidth)
{
  REAL fac = (REAL) (TWOPI / samprate);
  am->pll.freq.f = freq * fac;
  am->pll.freq.l = lofreq * fac;
  am->pll.freq.h = hifreq * fac;
  am->pll.phs = 0.0;
  am->pll.delay = cxJ;

  am->pll.iir.alpha = bandwidth * fac;                  /* arm filter */
  am->pll.alpha = am->pll.iir.alpha * 0.3f;             /* pll bandwidth */
  am->pll.beta = am->pll.alpha * am->pll.alpha * 0.25f;                 /* second order term */
  am->pll.fast_alpha = am->pll.alpha;
}

static void
pll (AMD am, COMPLEX sig)
{
  COMPLEX z = Cmplx ((REAL) cos (am->pll.phs), (REAL) sin (am->pll.phs));
  REAL diff;

  am->pll.delay.re = z.re * sig.re + z.im * sig.im;
  am->pll.delay.im = -z.im * sig.re + z.re * sig.im;
  diff = Cmag (sig) * ATAN2 (am->pll.delay.im, am->pll.delay.re);

  am->pll.freq.f += am->pll.beta * diff;

  if (am->pll.freq.f < am->pll.freq.l)
    am->pll.freq.f = am->pll.freq.l;
  if (am->pll.freq.f > am->pll.freq.h)
    am->pll.freq.f = am->pll.freq.h;
```

```
am->pll.phs += am->pll.freq.f + am->pll.alpha * diff;

  while (am->pll.phs >= TWOPI)
    am->pll.phs -= (REAL) TWOPI;
  while (am->pll.phs < 0)
    am->pll.phs += (REAL) TWOPI;
}

static REAL
dem (AMD am)
{
  am->lock.curr =
    (REAL) (0.999 * am->lock.curr + 0.001 * fabs (am->pll.delay.im));

  /* env collapse? */
  /* if ((am->lock.curr < 0.05) && (am->lock.prev >= 0.05)) {
     am->pll.alpha = 0.1 * am->pll.fast_alpha;
     am->pll.beta = am->pll.alpha * am->pll.alpha * 0.25;
     } else if ((am->pll.alpha > 0.05) && (am->lock.prev <= 0.05)) {
     am->pll.alpha = am->pll.fast_alpha;
     } */
  am->lock.prev = am->lock.curr;
  am->dc = 0.99f * am->dc + 0.01f * am->pll.delay.re;
  return am->pll.delay.re - am->dc;
}

/*---------------------------------------------------------------------------*/
/* public */
/*---------------------------------------------------------------------------*/

void
AMDemod (AMD am)
{
  int i;
  REAL demout;
  switch (am->mode)
    {
    case SAMdet:
      for (i = 0; i < am->size; i++)
                        {
                          pll (am, CXBdata (am->ibuf, i));
                          demout = dem (am);
                          CXBdata (am->obuf, i) = Cmplx (demout, demout);
                        }
```

```
break;
    case AMdet:
      for (i = 0; i < am->size; i++)
                        {
                          am->lock.curr = Cmag (CXBdata (am->ibuf, i));
                          am->dc = 0.9999f * am->dc + 0.0001f * am->lock.curr;
                          am->smooth = 0.5f * am->smooth + 0.5f * (am->lock.curr - am->dc);
                          /* demout = am->smooth; */
                          CXBdata (am->obuf, i) = Cmplx (am->smooth, am->smooth);
                        }
      break;
    }
}
AMD
newAMD (REAL samprate,
                        REAL f_initial,
                        REAL f_lobound,
                        REAL f_hibound,
                        REAL f_bandwid,
                        int size, COMPLEX * ivec, COMPLEX * ovec, AMMode mode, char *tag)
{
  AMD am = (AMD) safealloc (1, sizeof (AMDDesc), tag);
  am->size = size;
  am->ibuf = newCXB (size, ivec, tag);
  am->obuf = newCXB (size, ovec, tag);
  am->mode = mode;
  init_pll (am, samprate, f_initial, f_lobound, f_hibound, f_bandwid);
  am->lock.curr = 0.5;
  am->lock.prev = 1.0;
  am->dc = 0.0;
  return am;
}

void
delAMD (AMD am)
{
  if (am)
    {
      delCXB (am->ibuf);
      delCXB (am->obuf);
      safefree ((char *) am);
    }
}
```

Demodulador de AM escrito en lenguaje C

PowerSDR versión 1.8.0 (FLEX-RADIO)

Componentes de los equipos SDR

- Analógicos: atenuadores, amplificadores, filtros
- Dispositivos DSP: TI, Analog Devices
- Dispositivos ASIC
- Dispositivos programables FPGA

(Altera, Xilinx): miles de bloques funcionales programables, así como las conexiones entre ellos

Programación FPGA: lenguajes VHDL, Verilog.

- Conversores analógico a digital (ADC), digital a analógico (DAC): Analog Devices, Linear Technologies
- Conversores de velocidad digital (*Digital Down Converter, Digital Up Converter*)
- Controladores
- Interfaces (USB, RS232, FireWire...)
- Osciladores digitales, alimentación, etc.

Componentes de los equipos SDR: conversores ADC

Las características de la etapa de conversión A/D (linealidad, ruido térmico, precisión en voltaje y tiempo) influyen directamente en las del equipo SDR. Se le exigen:

- Velocidades de muestreo muy altas (superiores al doble de la máxima frecuencia de la señal a digitalizar)
- Número de bits de cuantización eficaces elevado
- Gran ancho de banda → uso de preselectores
- Gran margen dinámico
- Bajo coste

- 12 bit, 3,6 GHz (National)
- 16 bit, 250 MHz (Analog Devices)

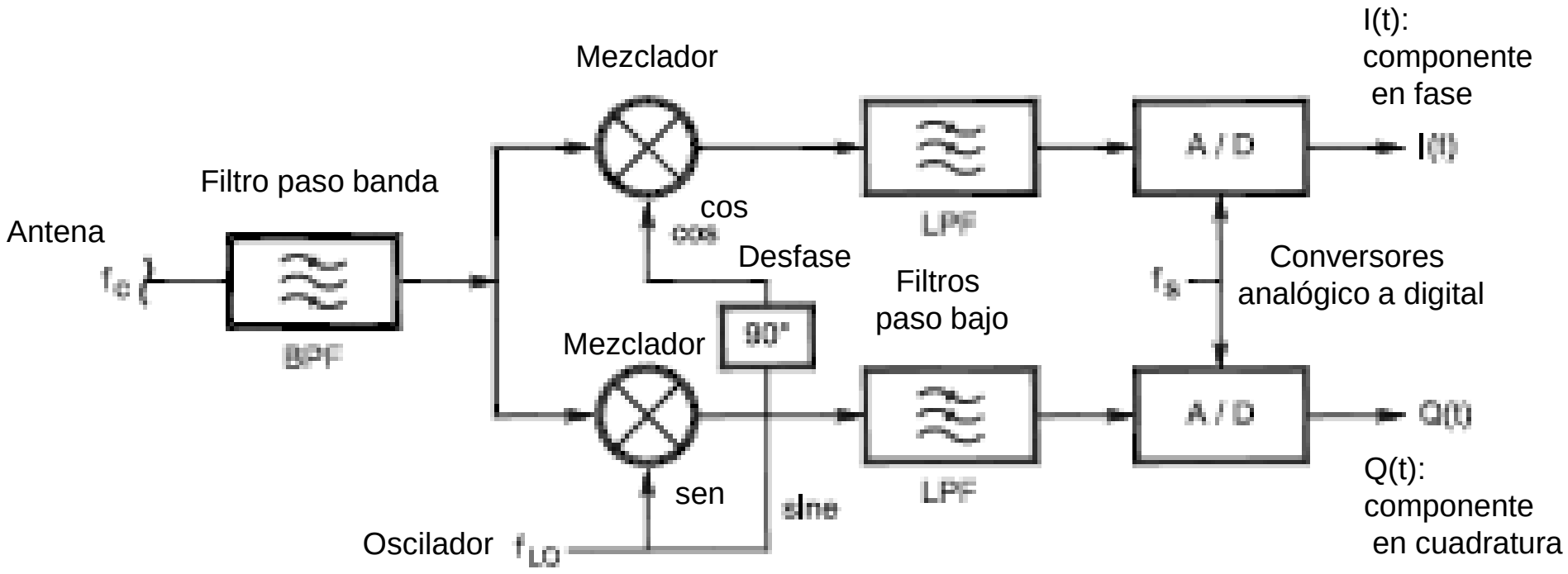
Tipos de equipos SDR

Equipos SDR para radioafición

- Equipos de conversión directa:
  SOFTROCK, FLEX-RADIO
- Equipos de digitalización directa:
  RFSPACE, PERSEUS, transceptor HPSDR

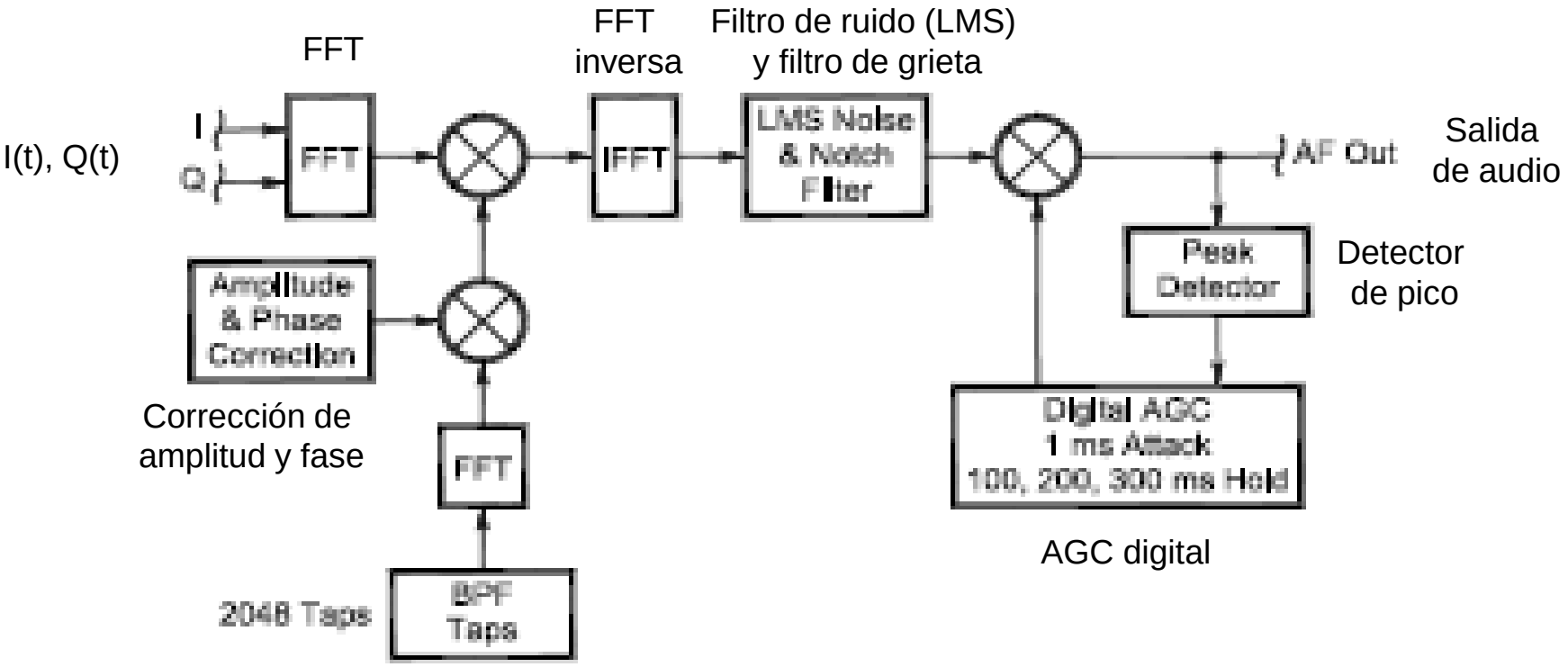
Equipos de conversión directa: parte *hardware*

- El equipo SDR, en recepción, entrega al ordenador el ancho de banda de interés en forma de dos señales llamadas señal en fase y señal en cuadratura (I y Q). Puede entregarlas analógicamente (tarjeta de sonido, caso del Softrock y el SDR-1000) o digitalmente (puertos USB, FireWire, Ethernet, etc.)
- En transmisión el proceso es el inverso

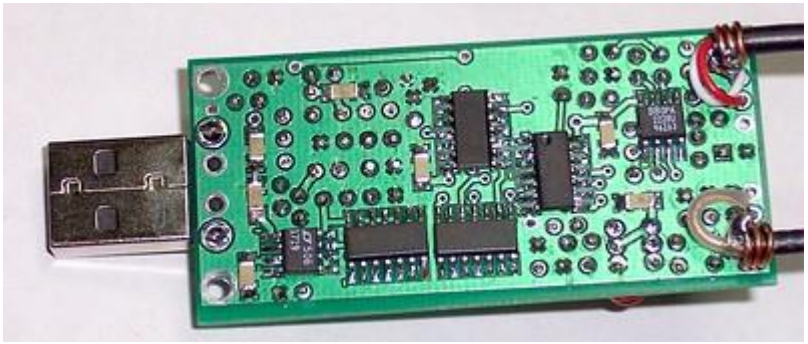


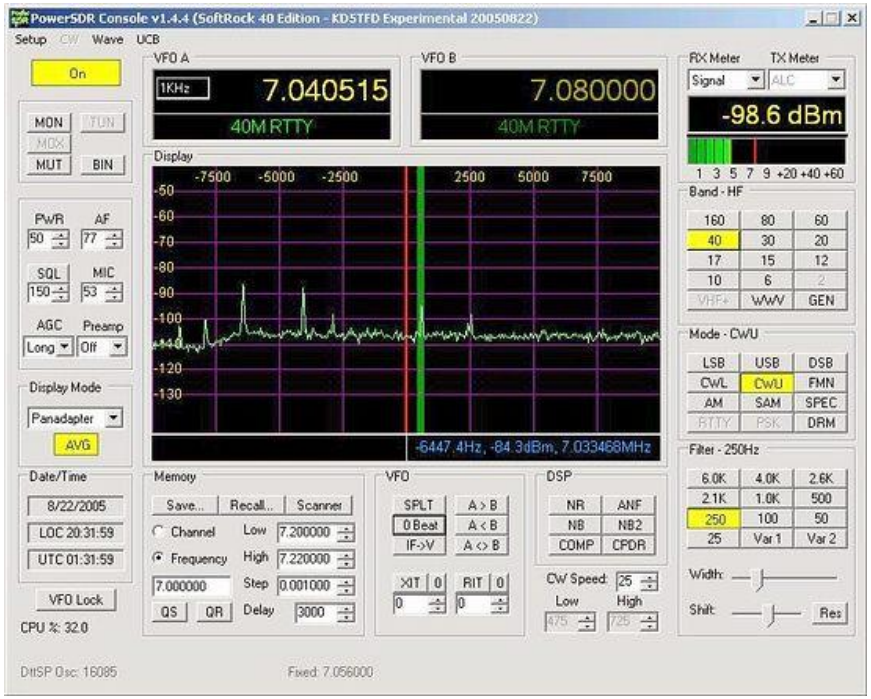
Equipos de conversión directa: parte *software*

- Un ordenador y/o un “chip” programable lleva a cabo las funciones de procesado de señal (filtrado, modulación/demodulación, reducción de ruido, AGC, análisis espectral, etc.) y otras como interfaz de usuario, control del equipo, etc.



Receptor SDR de conversión directa: SOFTROCK





Programa PowerSDR: apto para los receptores Softrock y otros SDR (incluidos transceptores)

Receptor SDR de conversión directa: SOFTROCK

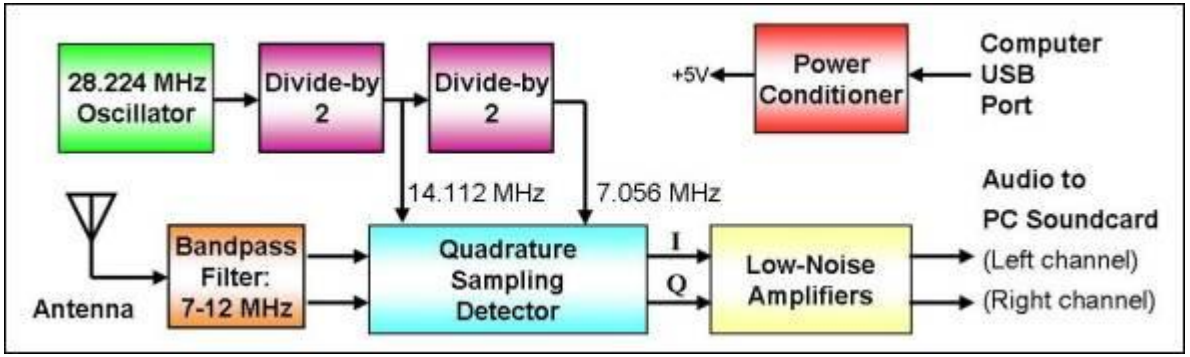
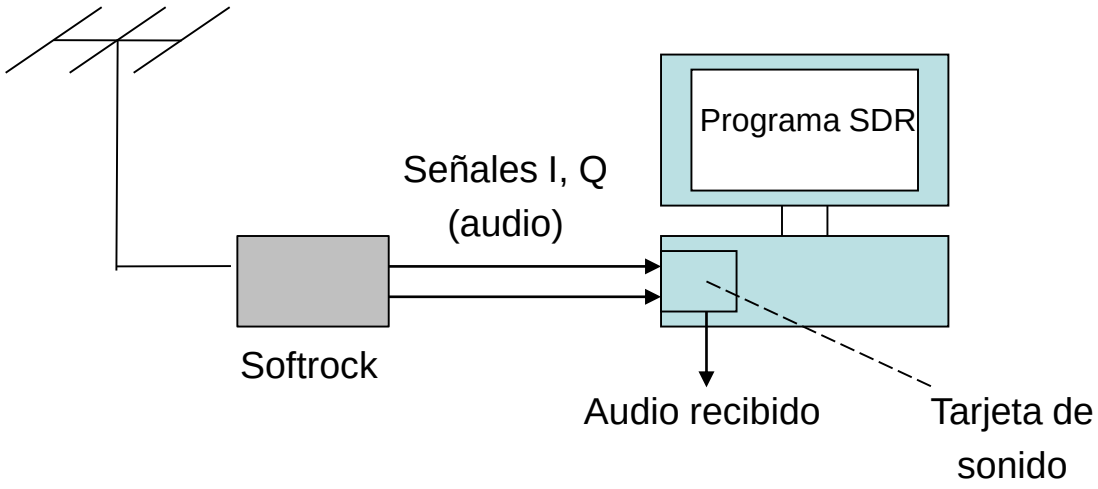
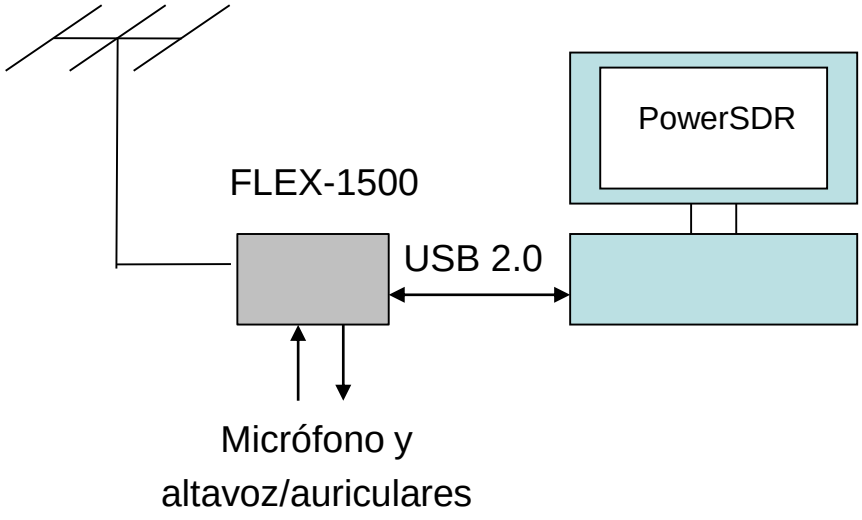


Diagrama de bloques de la versión para la banda de 40 metros



Transceptor SDR de conversión directa: FLEX-1500



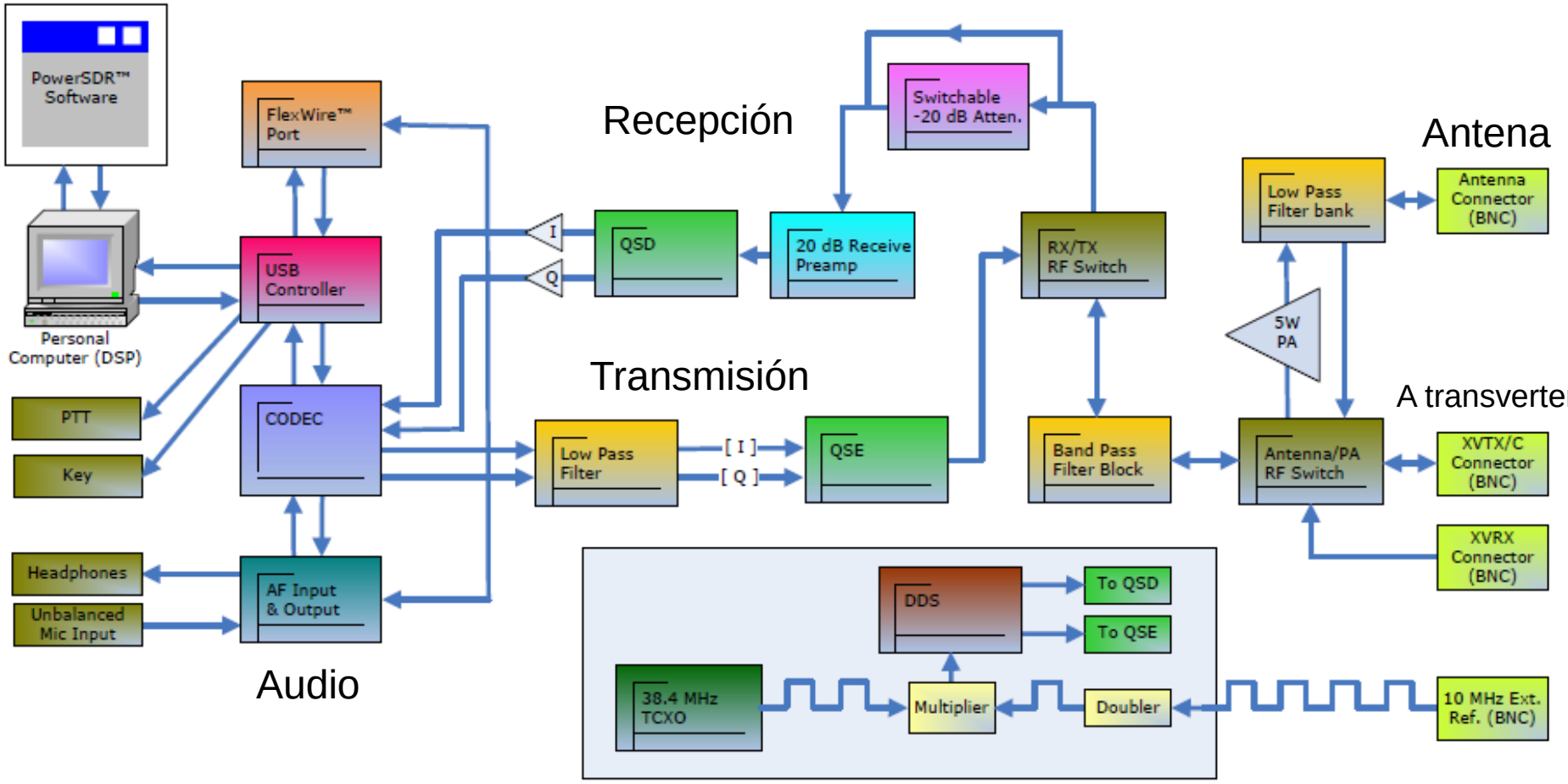
160 a 6 metros

RX: 0,01 a 54 MHz

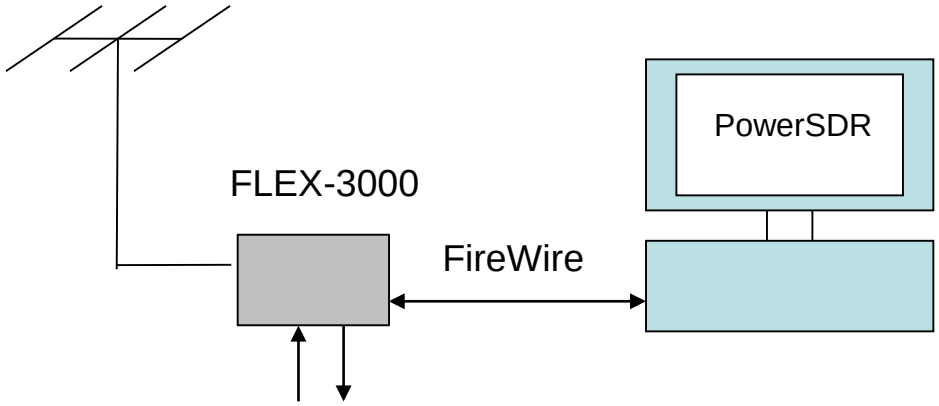
5 vatios RF

Transceptor SDR de conversión directa: FLEX-1500

Filtros RF: orden 4



Transceptor SDR de conversión directa: FLEX-3000



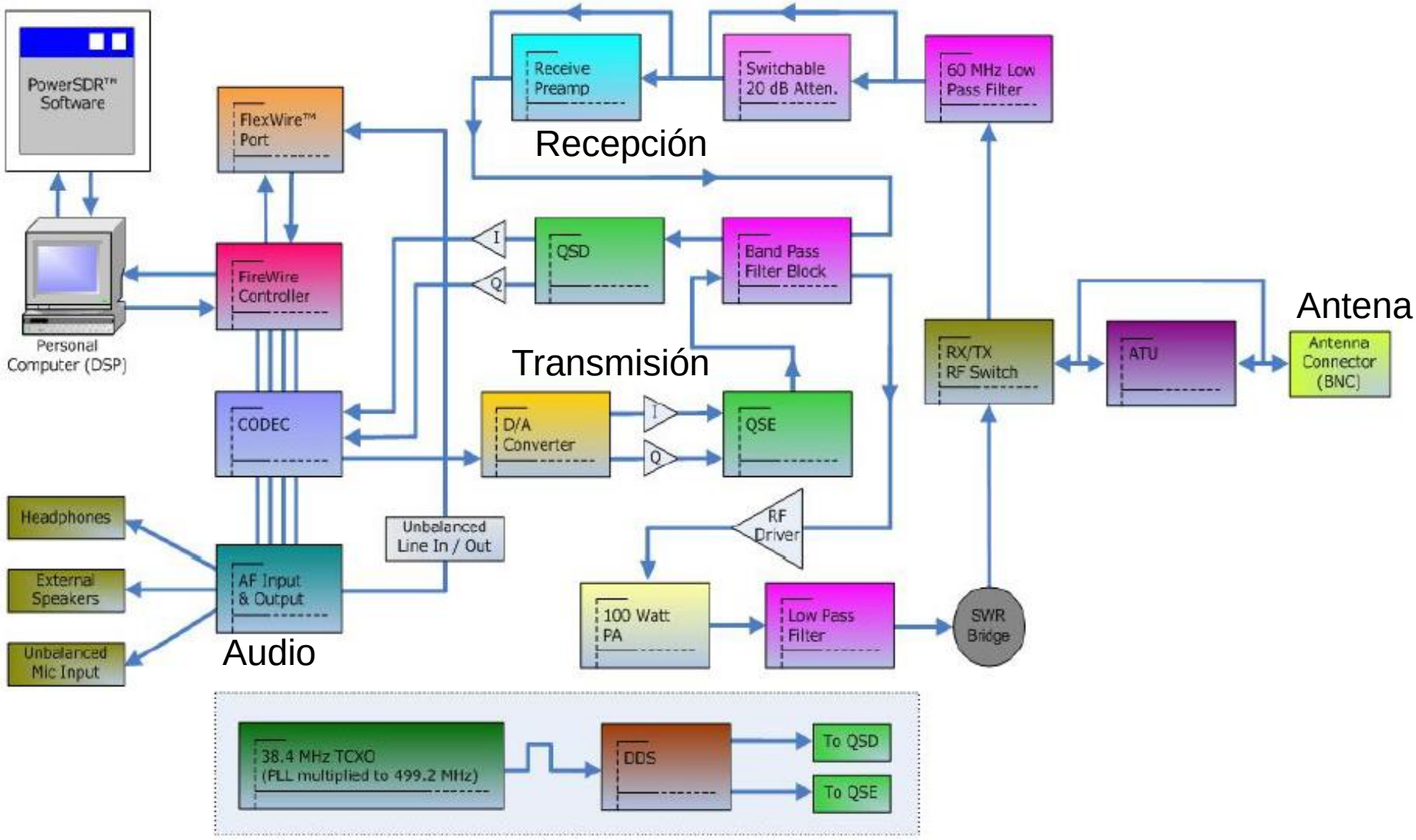
160 a 6 metros

RX: 0,01 a 65 MHz

100 vatios RF

Transceptor SDR de conversión directa: FLEX-3000

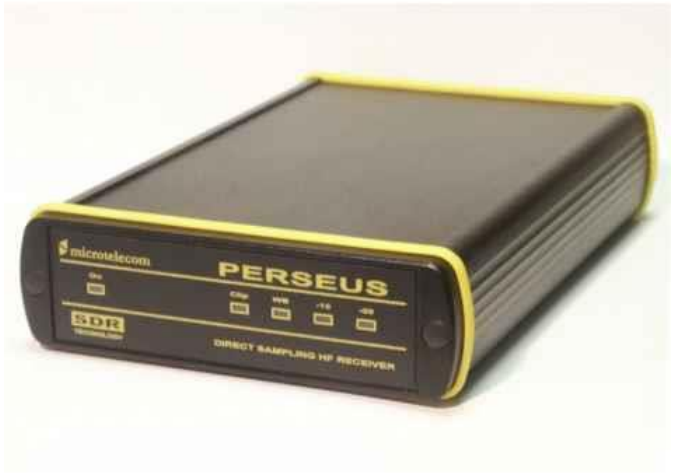
Filtros RF: orden 5
(orden 7 en 160 metros)



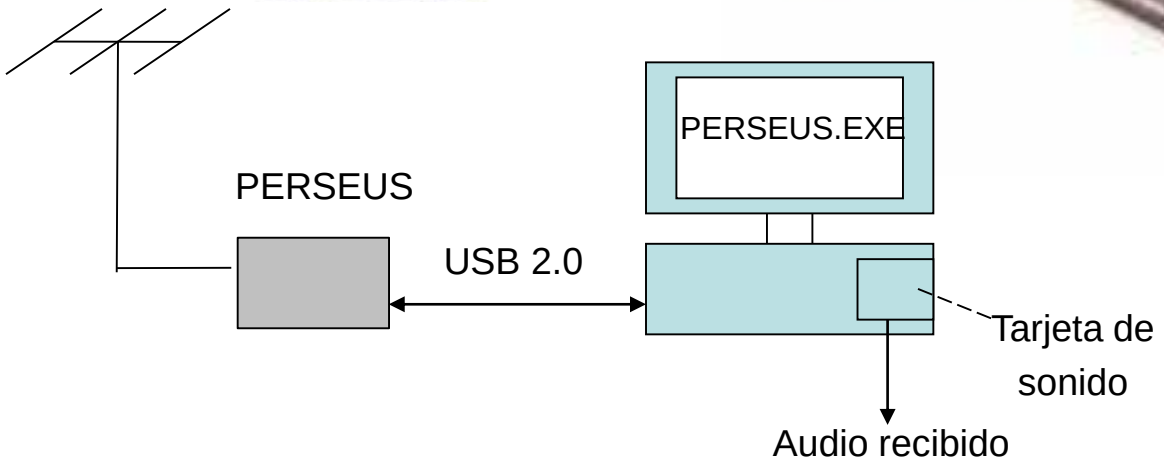
Equipos de conversión digital directa

- Muestreo directo (digitalización) de las señales de antena
- Preselección mediante filtros analógicos
- Se requiere una cierta amplificación
- Es posible la recepción en varias frecuencias simultáneamente
- Envían las señales I y Q al ordenador una vez digitalizadas
- Algunos requieren utilizar el dispositivo de sonido del ordenador como salida de audio recibido

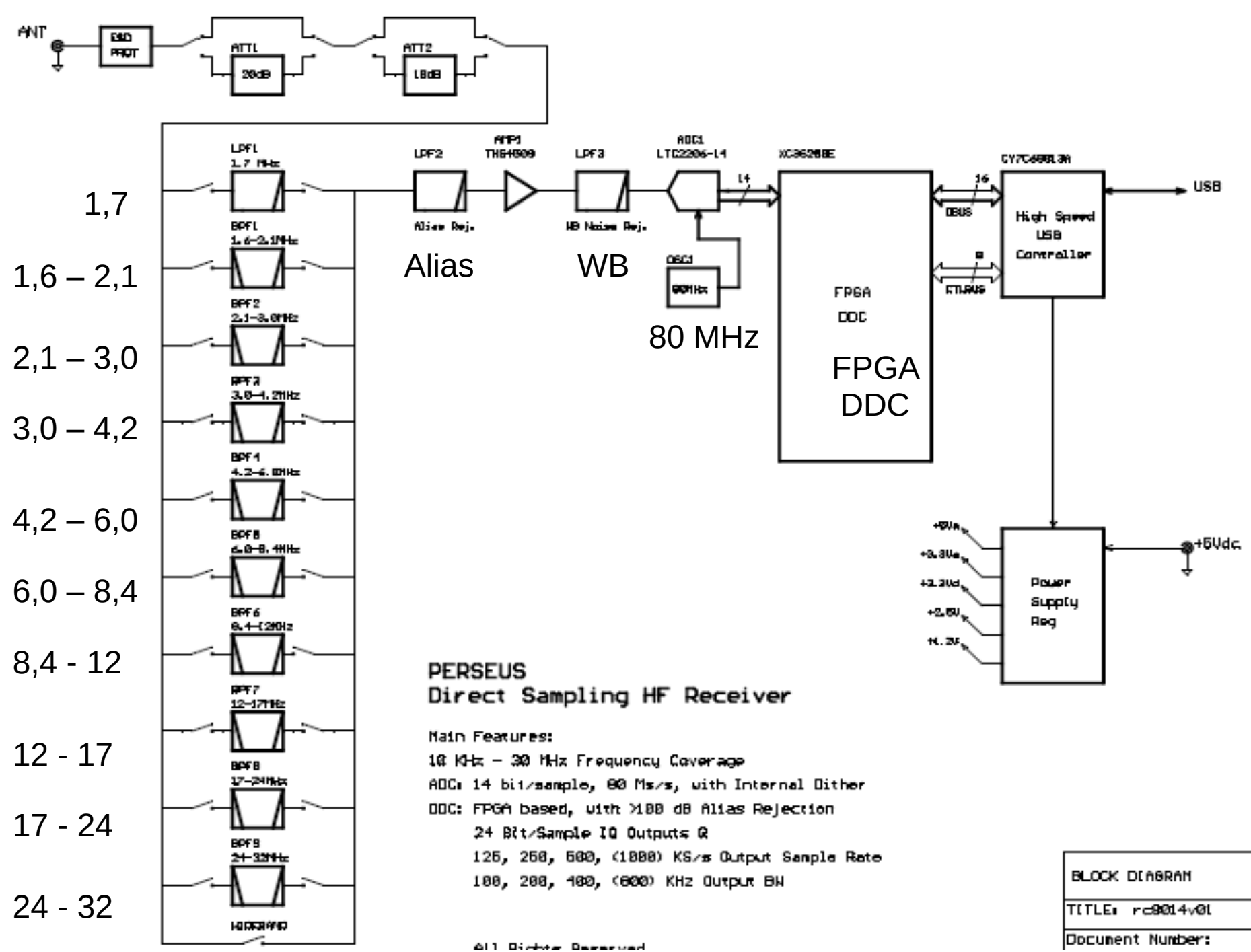
Receptor SDR de digitalización directa: Microtelecom PERSEUS



AstroRadio.com



0,01 MHz a 40 MHz
14 bits, 80 MHz



PERSEUS
Direct Sampling HF Receiver

Main Features:

10 KHz – 30 MHz Frequency Coverage

ADC: 14 bit/sample, 80 Ms/s, with Internal Dither

DDC: FPGA based, with >100 dB Alias Rejection

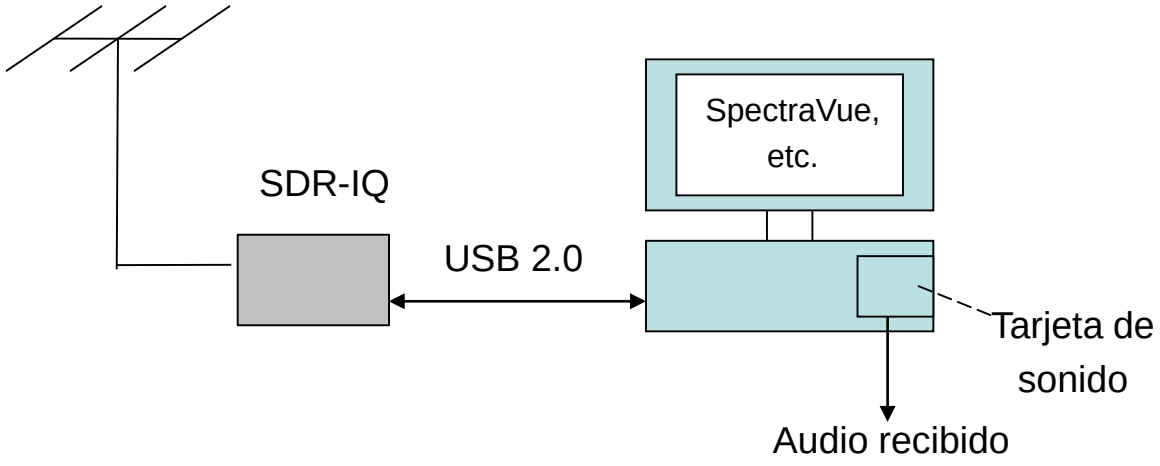
24 Bit/Sample IQ Outputs

125, 250, 500, (1000) KS/s Output Sample Rate

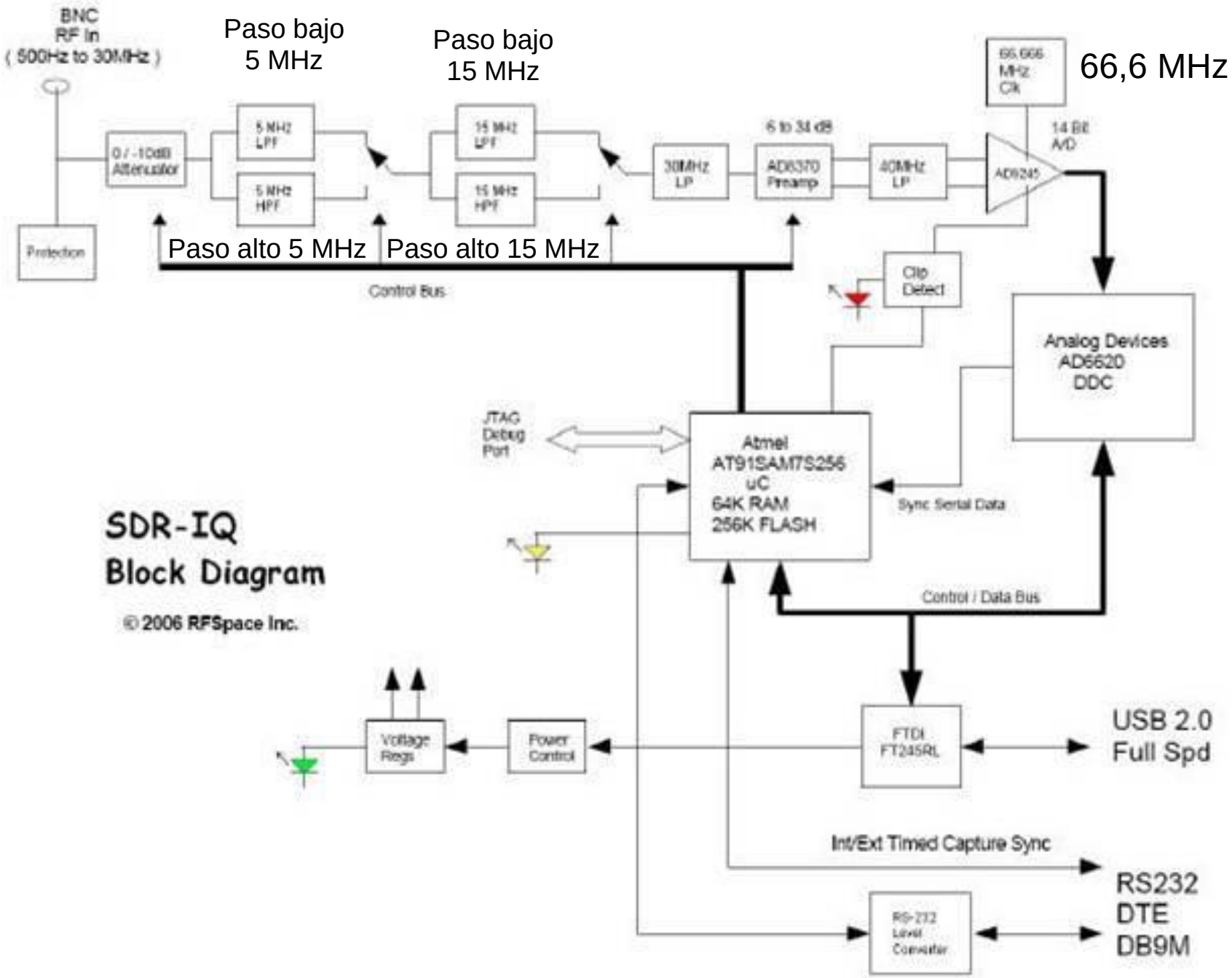
100, 200, 400, (800) KHz Output BW

BLOCK DIAGRAM
TITLE: rc9014v01
Document Number:

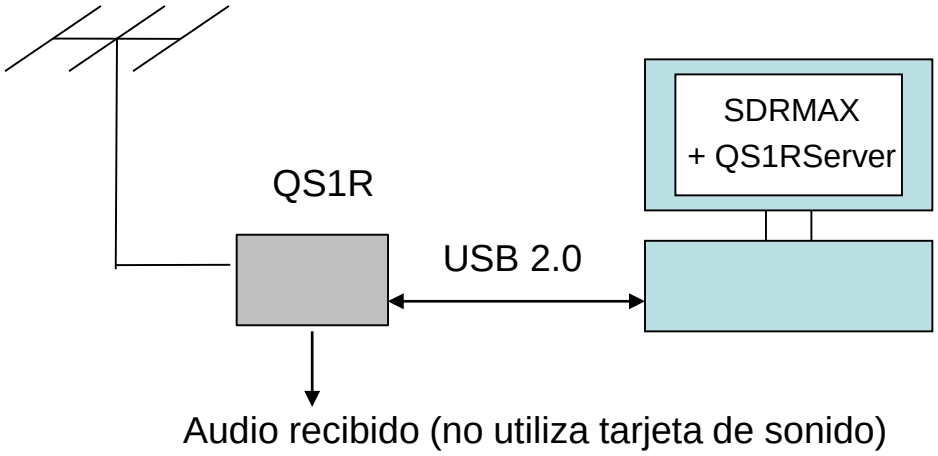
Receptor SDR de digitalización directa: RFSPACE SDR-IQ



500 Hz a 30 MHz
14 bits, 66,6 MHz


BNC
RF In
(500Hz to 30MHz)
Paso bajo
5 MHz
Paso bajo
15 MHz
66,6 MHz
66.666
MHz
Clk
0 / -10dB
Attenuator
5 MHz
LPF
5 MHz
HPF
15 MHz
LPF
15 MHz
HPF
30MHz
LP
6 to 34 dB
AD6370
Preamp
40MHz
LP
14 Bit
A/D
AD9245
Protection
Paso alto 5 MHz
Paso alto 15 MHz
Control Bus
Clip
Detect
Analog Devices
AD6620
DDC
JTAG
Debug
Port
Atmel
AT91SAM7S256
uC
64K RAM
256K FLASH
Sync Serial Data
SDR-IQ
Block Diagram
© 2006 RFSpace Inc.
Control / Data Bus
Voltage
Regs
Power
Control
FTDI
FT245RL
USB 2.0
Full Spd
Int/Ext Timed Capture Sync
RS232
DTE
DB9M
RS-232
Level
Converter

Receptor SDR de digitalización directa: QuickSilver QS1R



0,01 a 62,5 MHz

16 bits, 130 MHz

Entrada de banda ancha para recepción por submuestreo hasta 500 MHz (RevC)

Receptor SDR de digitalización directa: Microtelecom GEMINI

Doble receptor
28 a 170 MHz
14 bits, 200 MHz
Entrada de banda ancha para recepción por submuestreo hasta 500 MHz

El futuro

- Avances en electrónica:
  - Conversores ADC, DAC (mayor velocidad de digitalización, con más muestras/segundo y más bits/muestra): mejores prestaciones de los SDR en HF, y equipos SDR para V/UHF sin *transverters* (digitalización directa en V/UHF)
  - Mayor velocidad de procesado (menor tiempo de retardo)
- Avances en informática:
  - Mayor potencia (programas SDR más sofisticados, menor tiempo de retardo)
  - Conexiones más rápidas como Gigabit Ethernet, USB 3.0 a 5 Gbit/s: mayor ancho de banda a tratar por los SDR
  - Integración de microprocesadores en equipos SDR (no hará falta ordenador externo, ejemplo: FLEX-5000)
- Integración en los programas SDR de modos digitales nuevos y existentes
- Integración de los programas SDR en entornos multimedia
- Avances en telecomunicaciones:
  - Operación remota
  - Operación con dispositivos portátiles
  - Sincronización y procesado de las señales recibidas por SDR situados en distintas ubicaciones, sintonizados en una misma frecuencia, para obtener la señal óptima: GPS, internet
- Múltiples transmisores/receptores “virtuales” operando en un mismo equipo SDR
- Diversidad en recepción (procesado de señales recibidas de distintas antenas para obtener la mejor combinación posible)
- Ofrecerán la posibilidad de elegir adaptativamente modo, frecuencia y filtrado en función de las condiciones (equipo de radio cognitivo)
- Capacidad de detectar y eliminar interferencias de otras estaciones
- Incorporación de componentes microelectromecánicos específicos para radiofrecuencia (RF MEMS)

Javier, EA2DFH
http://radiodsp.blogspot.com/

Usuarios Flex-Radio habla hispana
http://es.groups.yahoo.com/group/FlexRadioEA/