DOCUMENTATION

Using Telnet LAN (Port 18)

Signal Generator Models
APSIN, APGEN, APSYN

Telnet provides a means of communicating with the signal generator over the LAN. The Telnet client, run on a LAN connected computer, will create a login session on the signal generator. A connection, established between computer and signal generator, generates a user interface display screen with "**>**" prompts on the command line.
Using the Telnet protocol to send commands to the signal generator is similar to communicating with the signal generator over LAN. You establish a connection with the signal generator and then send or receive information using predefined commands. Communication is interactive: one command at a time. The telnet service is available on **port 18**.
Once a telnet session to the device is established, the echo can be enabled by typing **SYST:COMM:SOCK:ECHO ON**
Following this command a prompt ">>" should become visible.

FREQuency Subsystem

Command	Parameters	Units
[SOURce]:FREQuency:[FIXed | CW]	fmin to fmax	Hz
[SOURce]:FREQuency:MODE	FIX|CW|SWEep|LIST| CHIR	FIXed
[SOURce]:FREQuency:STARt	fmin to fmax	Hz
[SOURce]:FREQuency:STOP	fmin to fmax	Hz
SOURce]:FREQuency:STEP[:LINear]?		Hz
[SOURce]:FREQuency:STEP:LOGarithmic?		

:FREQuency[:CW]
FREQuency <value><unit>
This command sets the signal generator output frequency.

FREQuency ?
This command ask the signal generator output frequency.

Exemple: > FREQ 900000000hz *pour configurer la fréquence de sortie du générateur à 900Mhz.*

OUTPut Subsystem

[:STATe] ON|OFF|1|0
:OUTPut[:STATe] ON|OFF|1|0
Turns RF output power on/off.

Discussion of the new Protocol and Packet format

RFPAnet commands and responses remain human interperable via simple terminal software. It is frequently a customer requirement to be able to use any of the controllers supplied to them with simple terminal software.
This precludes the use of Binary encoding and demands the use of ASCII characters. This is not the most efficient method in terms of packet size for moving data around, but the increased Baud rates of the new hardware developments mean that a much higher data throughput rate will still be achieved when compared with older RFPA controller designs. This project only demands infrequent changes and responses in a few milliSeconds are quite acceptable.
All packet data is represented in ASCII, (0-9), (a-z),(A-Z). There is no use of embedded control codes and no use of non terminal displayable characters.
The format offers greater flexibility and functional expandability than prior RFPA packet structures. (This project is NOT RFPAnet2 standard compatible, some simplification was possible, but the packet structure remains “in the style of’ RFPAnet2”)

NOTE: all packets are fixed length, all packet components are also fixed length and must be leading zero packed in the case of numeric data or leading character packed in the case of Alpha data (use ascii dec 32 = ‘space’ is preferred, other characters may cause confusion to local rack LCD)

Name	**Bytes**	**Purpose**	**Content, always ASCII and comments**
Product Identifier	2	Header to allow easy identification of start of packet	Alpha text range ‘aa’ to ‘zz’
Command	2	Contains the instruction to the Controller/Server	Numeric range 00-99
Path selection	1	Address of individual path or channel	Numeric range 0-6
Data**	5	To pass data associated with the command to the Server	Numeric range 00000-65535 5 characters

**** DATA FORMAT**.
Data field will contain data in multiple formats depending on Command.
Representation of Numeric data is in the following format:
Five ASCII digits (decimal, ex 1= asc 49) ……. Integer number in the range 00000 to 65535

Packet Structure Quick Guide.
NOTE: all packets are fixed length, all packet components are also fixed length and numeric data fields must be leading zero packed and character data fields must be character packed to the appropriate length.
** the “one command response outstanding at any time” rule must be observed. **

Packet Explanation (spaces have been added between fields to clarify structure, they must not be included in the final packet)

aa 00 0 00000 (total packet, spaces added for clarity, do not include in real packets)

aa= product identifier (will be 'rf' this project, IS case sensitive)

00= command (look in instruction table for descriptions)

0=path selection (for RF paths/channels, '0' is reserved for server address= main)

00000= 5 digit integer,

so finally we get
rf00000000

Command range 00-xx
NOTE: all server responses contain either an ‘echo’ of the command field, or the data requested.
‘path selection’ for commands ‘00’ & ‘53’ is “don’t care” but must be present in the packet , best to use ‘0’

Real world examples are contained elsewhere in this document.
packets must be sent with fixed length and *no* terminator(s) (*for example no cr, lf is necessary*)

Cmd.	Function	Data field expected	Data field response (can also be error code)
50	Set Channel RF level **channel number in Field Path Selection , 1-6 valid.	Type: Numeric(0) Content:5 Digits, leading zero packed,valid values; 00000 to 04095	Echo of that sent
51	Store current value of Channel to EEPROM **channel number in Field Path Selection , 1-6 valid.	Type: Numeric(0) Content: Don’t care	Echo of that sent
52	Send me Channel RF level **channel number in Field Path Selection , 1-6 valid.	Type: Numeric(0) Content: Don’t care	5 Digits, leading zero packed. Last 3 digits are Power Level in Range; 000 to 999 mW Example: 000985 = 985 mW

edit this section for this controller
rf50101856 << command 'set level of channel 1 to 1856 ' packet _sent_ to amp, *no* cr,lf necessary
rf50101856 << response from server **with** cr and lf....

Structure de la trame

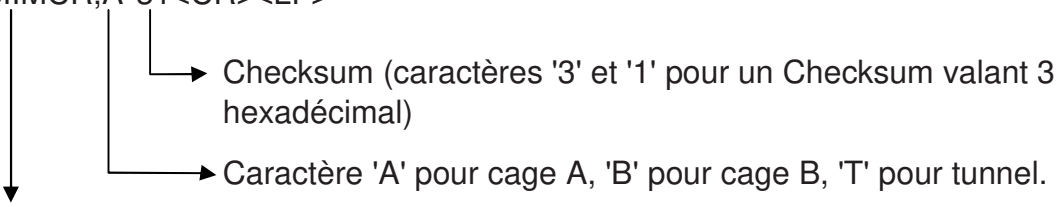
Champ de la trame	Signification
'$'	HEX24 – entête de trame.
<Adress field>	champ TALKER qui indique qui envoie la trame et spécifie son contenu.
<Data field> [","<data field>]….[","<data field>]	champs DATA. De 0 à N champs, tous séparés par des virgules.
"*"<checksum field>	champ Checksum, précédé du caractère '*'
<CR><LF>	HEX 0D 0A- fin de trame

Le champ "Checksum".

Le champ Checksum, constitué de deux caractères, est codé sur un octet. Le Checksum est obtenu en opérant un ou exclusif entre tous les caractères compris entre le caractère '$' et le caractère '*' ('$' et '*' étant exclus du calcul).

- ***Contenu de la trame propriétaire informant de la position du rat:***

$IIMCR,A*31<CR><LF>



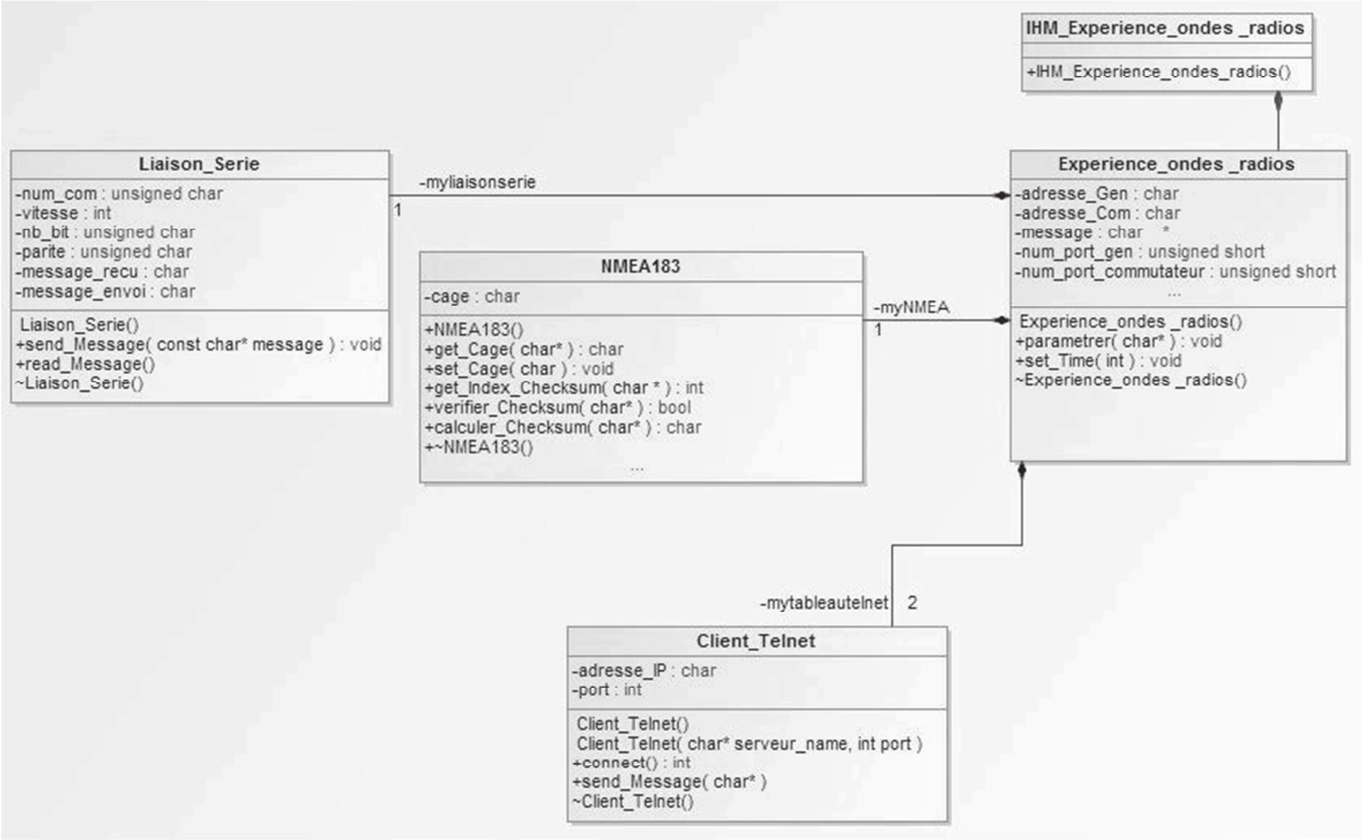
"II" pour trame générique, "MCR" pour indiquer que le contenu de la trame est relatif à la position du rat.

Extrait de la table ASCII

Ce tableau ne contient que les valeurs utiles de la table ASCII pour cette étude.

Décimal	Hexadécimal	Caractére
36	24	**$**
42	2A	*
44	2C	,
65	41	**A**
66	42	**B**
67	43	**C**
73	49	**I**
77	4D	**M**
82	52	**R**

Diagramme de classes partiel de l'application.



Documentation PP5 Méthode `substr`

std::string::substr

```
string substr (size_t pos = 0, size_t len = npos) const;
```

Generate substring

Returns a newly constructed string object with its value initialized to a copy of a substring of this object.

The substring is the portion of the object that starts at character position *pos* and spans *len* characters (or until the end of the string, whichever comes first).

PARAMETERS

pos Position of the first character to be copied as a substring.
If this is equal to the *string length*, the function returns an empty string.
If this is greater than the *string length*, it throws out_of_range.
Note: The first character is denoted by a value of 0 (not 1).

len Number of characters to include in the substring (if the string is shorter, as many characters as possible are used).
A value of string::npos indicates all characters until the end of the string.

size_t is an unsigned integral type (the same as member type string::size_type).

RETURN VALUE

A string object with a substring of this object.

Exemple

```
1  // string::substr
2  #include <iostream>
3  #include <string>
4
5  int main ()
6  {
7    std::string str="We think in generalities, but we live in details.";
8                                             // (quoting Alfred N. Whitehead)
9
10   std::string str2 = str.substr (3,5);     // "think"
11
12   std::size_t pos = str.find("live");      // position of "live" in str
13
14   std::string str3 = str.substr (pos);     // get from "live" to the end
15
16   std::cout << str2 << ' ' << str3 << '\n';
17
18   return 0;
19 }
```

Documentation PP6 Fonction `stoi ()`

`std::stoi`

```
int stoi (const string&  str, size_t* idx = 0, int base = 10);
```

Convert string to integer

Parses *str* interpreting its content as an integral number of the specified *base*, which is returned as an int value.

If *idx* is not a null pointer, the function also sets the value of *idx* to the position of the first character in *str* after the number.

PARAMETERS

str String object with the representation of an integral number.

idx Pointer to an object of type size_t, whose value is set by the function to position of the next character in *str* after the numerical value.
This parameter can also be a null pointer, in which case it is not used.

base Numerical base (radix) that determines the valid characters and their interpretation.
If this is 0, the base used is determined by the format in the sequence (see strtol for details). Notice that by default this argument is 10 (base 10), not 0.

PARAMETERS

A string object with a substring of this object.

Extrait de : `http://www.cplusplus.com/reference/string/stoi/`

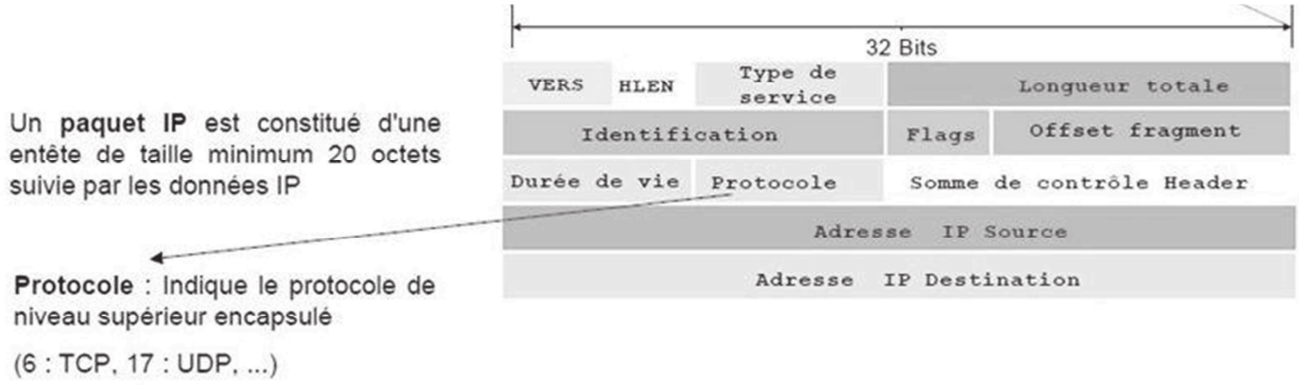
```
// stoi example
#include <iostream>   // std::cout
#include <string>     // std::string, std::stoi

int main ()
{
  std::string str_dec = "2001, A Space Odyssey";
  std::string str_hex = "40c3";
  std::string str_bin = "-10010110001";

  std::string::size_type sz;   // alias of size_t

  int i_dec = std::stoi (str_dec,&sz);
  int i_hex = std::stoi (str_hex,nullptr,16);
  int i_bin = std::stoi (str_bin,nullptr,2);

  std::cout << str_dec << ": " << i_dec << " and [" << str_dec.substr(sz) << "]\n";
  std::cout << str_hex << ": " << i_hex << '\n';
  std::cout << str_bin << ": " << i_bin << '\n';

  return 0;
}
```

Principe d'encapsulation et format de l'en-tête Ethernet :

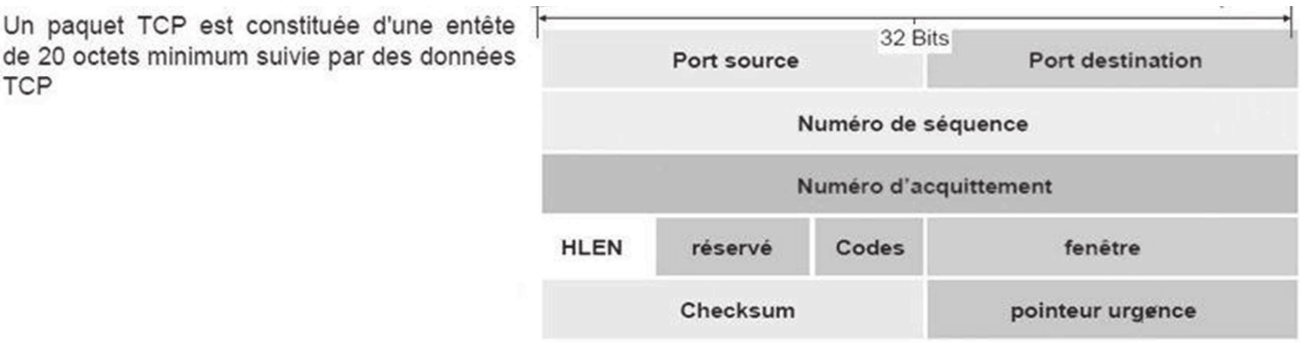
En-tête ETHERNET						
6 octets	*6 octets*	*2 octets*				
@ destination	**@ source**	**Type** *(0x0800 pour IP)*	**En-tête IP**	**En-tête TCP**	**Données**	**CRC**

Le préambule de la trame Ethernet n'est pas représenté par souci de clarté

Format de l'en-tête IP (format minimum de 20 octets) :



Format de l'en-tête TCP (format minimum de 20 octets) :



Documentation PP8 Principales requêtes SQL

Créer une base de données :	`create database nom_de_la_base;`
Supprimer une base de données	`drop database nom_de_la_base;`

Créer une table dans la base de données active	`create table nomTable (`
	`  id int auto_increment,`
	`  champ1 double,`
	`  champ2 float,`
	`  champ3 varchar,`
	`  champ4 timestamp,`
	`  champ5 boolean default false,`
	`  ...,`
	`  primary key(id));`

Lister la structure d'une table	`describe nomTable;`
Sélectionner toutes les informations de la table	`select * from nomTable;`
Sélectionner seulement les informations d'un champ	`select nomChamp from nomTable;`

Sélectionner tous les champs de la table nomTable correspondant à deux critères.	`select * from nomTable`
	`  where nomChamp1 = 'poste'`
	`  and nomChamp3 < 12;`

Sélectionner sur plusieurs tables (jointure)
`nomTable1.nomChamp1 est clé primaire.`
`nomTable2.nomChamp4 est une clé étrangère vers nomTable1.`
	`select * from nomTable1, nomTable2`
	`  where nom_table1.nomChamp1 =`
	`  nom_table2.nomChamp4;`

Écrire une nouvelle entrée dans une table	`insert into nomTable(champ1,champ2)` `  values(32.327432, 'un texte');`

Modifier les informations d'un enregistrement dont le champ date = '2018/07/21 0:28:12';	`update nomTable set`
	`  nomChamp1 = 10,`
	`  valeur2 = 32`
	`  where date = '2018/07/21 0:28:12';`

Ajouter des nouveaux champs (colonnes) dans une table	`alter table nomTable`
	`  add champ1 double,`
	`  add champ2 boolean default false;`

Écrire une nouvelle heure (heure actuelle grâce à l'appel de la fonction now()) dans un champ de type timestamp	`insert into nomTable(champ1)` `  values(now());`

F2250NLGK Datasheet

Voltage Variable RF Attenuator

50MHz to 6000MHz

General Description

The F2250 is a low insertion loss **V**oltage **V**ariable RF **A**ttenuator (VVA) designed for a multitude of wireless and other RF applications. This device covers a broad frequency range from 50MHz to 6000MHz. In addition to providing low insertion loss, the F2250 provides excellent linearity performance over its entire voltage control and attenuation range.

The F2250 uses a single positive supply voltage of 3.15V to 5.25V. Other features include the V_{MODE} pin allowing either positive or negative voltage control slope vs attenuation and multi-directional operation meaning the RF input can be applied to either RF1 or RF2 pins. Control voltage ranges from 0V to 3.6V using either positive or negative control voltage slope.

Competitive Advantage

F2250 provides extremely low insertion loss and superb IP3, IP2, Return Loss and Slope Linearity across the control range. Comparing to the previous state-of-the-art for silicon VVAs this device is better as follows:

- ✓ Insertion Loss @ 2000MHz: 1.4dB vs. 2.8dB
- ✓ Insertion Loss @ 6000MHz: 2.7dB vs. 7dB
- ✓ Maximum Attenuation Slope: 33dB/Volt vs. 53dB/Volt
- ✓ Minimum Return Loss up to 6000MHz: 12.5dB vs. 7dB
- ✓ Minimum Output IP3: 31dBm vs. 15dBm
- ✓ Minimum Input IP2: 87dBm vs. 80dBm
- ✓ Maximum Operating Temperature: +105°C vs. +85°C

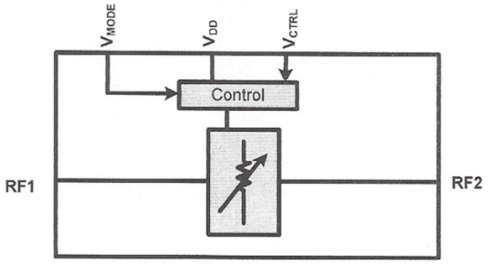
Applications

- Base Station 2G, 3G, 4G
- Portable Wireless
- Repeaters and E911 systems
- Digital Pre-Distortion
- Point to Point Infrastructure
- Public Safety Infrastructure
- WIMAX Receivers and Transmitters
- Military Systems, JTRS radios
- RFID handheld and portable readers
- Cable Infrastructure
- Wireless LAN
- Test / ATE Equipment

Features

- Low Insertion Loss: 1.4dB @ 2000MHz
- Typical / Min IIP3: 65dBm / 47dBm
- Typical / Min IIP2: 95dBm / 87dBm
- 33.6dB Attenuation Range
- Bi-directional RF ports
- +34.4dBm Input P1dB compression
- V_{MODE} pin allows either positive or negative attenuation control response
- Linear-in-dB attenuation characteristic
- Supply voltage: 3.15V to 5.25V
- V_{CTRL} range: 0V to 3.6V using 5V supply
- +105°C max operating temperature
- 3mm x 3mm, 16-pin QFN package

Device Block Diagram



Ordering Information

Part# Matrix

Part#	RF Freq Range (MHz)	Insertion Loss (dB)	IIP3 (dBm)	Pinout Compatibility
F2250	**50 - 6000**	**1.4 (at 2GHz)**	**+65**	**RFMD**
F2255	1 - 3000	1.1 (at 500MHz)	+60	
F2258	50 - 6000	1.4 (at 2GHz)	+65	Hittite

KATHREIN
Antennen · Electronic

Rosenheim, 21.11.2006
MS/AG

Customer Information :

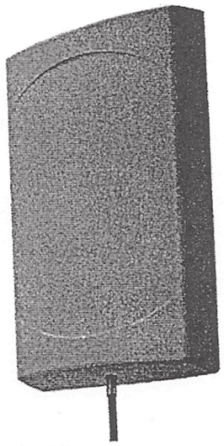
Product Replacement : 800 10248 → 800 10465

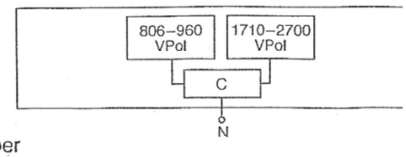
We are pleased to inform you about the new development of the vertical polarized multi-band antenna **800 10465** for indoor use, with launch date April 2007.

This newly developed antenna 800 10465 will replace 800 10248, which will be phased-out from July 1st 2007. The advantage of the new antenna is the extended frequency range from 1710-2500 MHz to 1710-2700 MHz.

Details about the electrical and mechanical specification are listed in the table below:

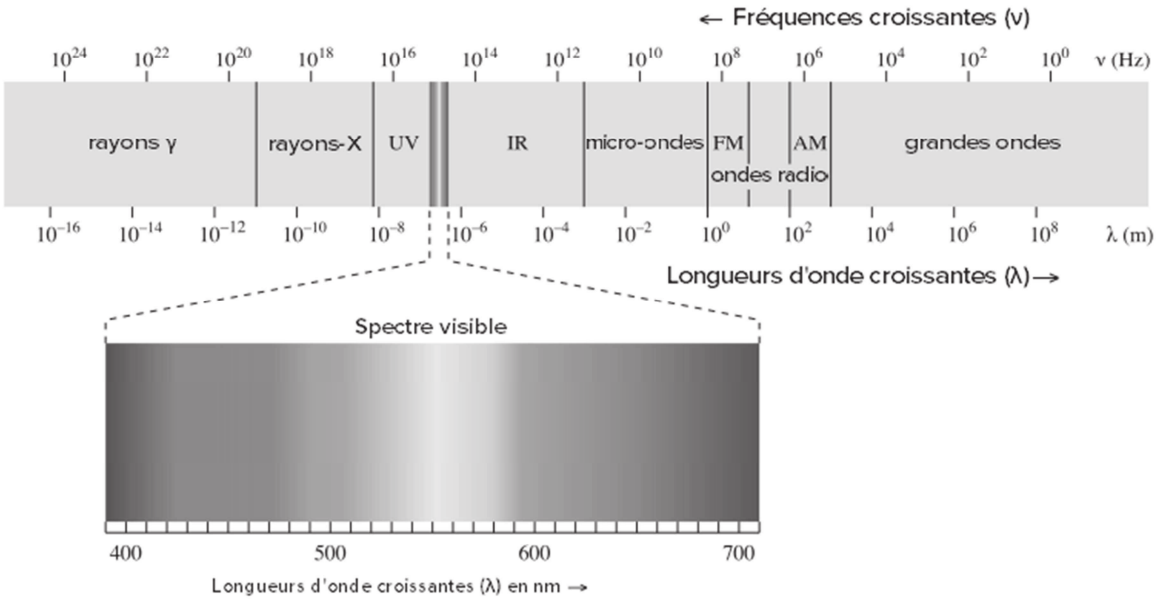
Type No.	**800 10465**	800 10248
Availability	**April 2007**	phase-out from July 1st 2007
Frequency Range	**806 – 960 MHz / 1710 – 2700 MHz**	806 – 960 MHz / 1710 – 2500 MHz
Polarization	**Vertical**	Vertical
Gain	**Approx. 7 dBi**	Approx. 7 dBi
Half-power beam width	**Horizontal: Approx. 90°**	Horizontal: Approx. 90°
Impedance	**50 Ohm**	50 Ohm
VSWR	**806 – 960 MHz: < 2.0 1710 – 2200 MHz: < 2.0 2200 – 2400 MHz: < 2.5 2400 – 2700 MHz: < 2.0**	806 – 960 MHz: < 2.0 1710 – 2200 MHz: < 2.0 2200 – 2400 MHz: < 2.5 2400 – 2500 MHz: < 2.0
Max. power	**50 W (at 50 °C ambient temp.)**	50 W (at 50 °C ambient temp.)
Input	**Cable RG 223/CU of 1 m length, white, with N female connector**	Cable RG 223/CU of 1 m length, white, with N female connector
Weight	**500 g**	500 g
Packing size	**363 x 152 x 62 mm**	363 x 152 x 62 mm
Height/width/depth	**231 / 140 / 50 mm**	231 / 140 / 50 mm





Peter Scholz
Head of Sales and Technical Marketing
Mobile Communication Systems
Kathrein-Werke KG

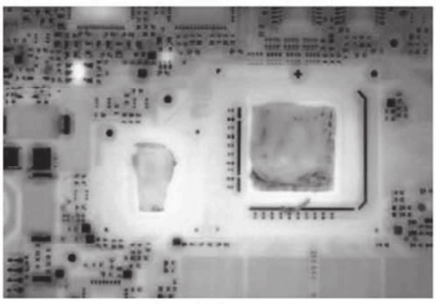
Anton Guggenhuber
Sales and Technical Marketing
Mobile Communication Systems
Kathrein-Werke KG

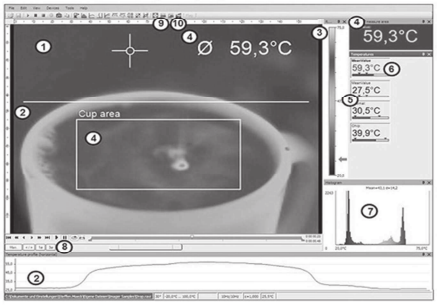

← Fréquences croissantes (ν)
10^{24} 10^{22} 10^{20} 10^{18} 10^{16} 10^{14} 10^{12} 10^{10} 10^{8} 10^{6} 10^{4} 10^{2} 10^{0} ν (Hz)
rayons γ
rayons-X
UV
IR
micro-ondes
FM
AM
ondes radio
grandes ondes
10^{-16} 10^{-14} 10^{-12} 10^{-10} 10^{-8} 10^{-6} 10^{-4} 10^{-2} 10^{0} 10^{2} 10^{4} 10^{6} 10^{8} λ (m)
Longueurs d'onde croissantes (λ) →
Spectre visible
400
500
600
700
Longueurs d'onde croissantes (λ) en nm →

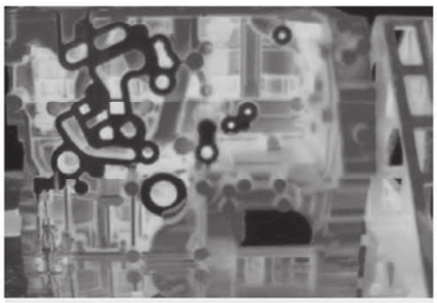
Compact high resolution infrared camera

Features:

- Smallest camera in its class 46 x 56 x 76 - 100 mm
- Interchangeable lenses and industrial accessories
- Switchable Framerate 80Hz/27Hz
- License-free analysis software and full SDK included







For further information as well als the product configurator, please visit www.optris.global/

Technical specifications	
Optical resolution	382 x 288 pixels
Detector	FPA, uncooled (25 µm x 25 µm)
Spectral range	8 - 14 µm
Temperature range	–20 ... 100 °C, 0 ... 250 °C, (20) 150 ... 900 °C[1], optional temperature range: 200 ... 1500 °C[2]
Frame rate	Switchable 27Hz/80Hz
Optik (FOV)	13° x 10° / f = 41 mm or 29° x 22° / f = 18.7 mm or 53° x 40° / f = 10.5 mm or 80° x 56° / f = 7.7 mm
Thermal sensitivity (NETD)	75 mK with 29° x 22° FOV / F = 0.8 75 mK with 53° x 40° FOV / F = 0.8 75 mK with 80° x 56° FOV / F = 0.8 0.1 K with 13° x 10° FOV / F = 1.0
Accuracy	±2 °C or ±2 %, whichever is greater
PC interfaces	USB 2.0 / optional USB to GigE (PoE) interface
Standard process interface (PIF)	0 – 10 V input, digital input (max. 24 V), 0 – 10 V output
Industrial process interface (PIF)	2x 0 – 10 V input, digital input (max. 24 V), 3x 0 – 10 V output, 3x relays (0–30 V/ 400 mA), fail-safe relay
Cable length (USB)	1 m (standard), 5 m, 10 m, 20 m 5 m and 10 m also as high temperature USB cable (180 or 250 °C)
Ambient temperature	0 ... 50 °C
Storage temperature	– 40 ... 70 °C
Relative humidity	20 – 80 %, non condensing
Enclosure (size / protection)	46 x 56 x 76 - 100 mm (depending on lens + focus position) / IP 67 (NEMA 4)
Weight	320 g, incl. lens
Shock / Vibration[3]	IEC 60068-2
Tripod mount	¼ – 20 UNC
Power supply	via USB
Scope of supply (standard)	• USB camera, incl. 1 lens • USB cable (1 m) • Table tripod • PIF cable, incl. terminal block (1 m), • Software package optris PIX Connect, • Aluminum case

[1] Accuracy effective starting at 150 °C
[2] Not available for 80° optics
[3] For further details see operator's manual