



INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

G.191

(11/2000)

SERIES G: TRANSMISSION SYSTEMS AND MEDIA,
DIGITAL SYSTEMS AND NETWORKS

International telephone connections and circuits –
Software tools for transmission systems

Software tools for speech and audio coding standardization

ITU-T Recommendation G.191

(Formerly CCITT Recommendation)

ITU-T G-SERIES RECOMMENDATIONS

TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

INTERNATIONAL TELEPHONE CONNECTIONS AND CIRCUITS	G.100–G.199
General definitions	G.100–G.109
General Recommendations on the transmission quality for an entire international telephone connection	G.110–G.119
General characteristics of national systems forming part of international connections	G.120–G.129
General characteristics of the 4-wire chain formed by the international circuits and national extension circuits	G.130–G.139
General characteristics of the 4-wire chain of international circuits; international transit	G.140–G.149
General characteristics of international telephone circuits and national extension circuits	G.150–G.159
Apparatus associated with long-distance telephone circuits	G.160–G.169
Transmission plan aspects of special circuits and connections using the international telephone connection network	G.170–G.179
Protection and restoration of transmission systems	G.180–G.189
Software tools for transmission systems	G.190–G.199
GENERAL CHARACTERISTICS COMMON TO ALL ANALOGUE CARRIER-TRANSMISSION SYSTEMS	G.200–G.299
INDIVIDUAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON METALLIC LINES	G.300–G.399
GENERAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON RADIO-RELAY OR SATELLITE LINKS AND INTERCONNECTION WITH METALLIC LINES	G.400–G.449
COORDINATION OF RADIOTELEPHONY AND LINE TELEPHONY	G.450–G.499
TESTING EQUIPMENTS	G.500–G.599
TRANSMISSION MEDIA CHARACTERISTICS	G.600–G.699
DIGITAL TERMINAL EQUIPMENTS	G.700–G.799
DIGITAL NETWORKS	G.800–G.899
DIGITAL SECTIONS AND DIGITAL LINE SYSTEM	G.900–G.999

For further details, please refer to the list of ITU-T Recommendations.

Software tools for speech and audio coding standardization

Summary

G.191 Annex A describes the ITU-T Software Tools Library release 2000 (STL2000). The STL2000 contains improvements and error corrections of the STL96 (the version of the ITU-T STL approved in 1996). In particular, a G.727 module, new filter types, and a Basic Operator module have been added, as well as support for more modern compilers.

This Recommendation includes an electronic attachment containing STL2000 Software Tool Library and manual.

Source

ITU-T Recommendation G.191 was revised by ITU-T Study Group 16 (2001-2004) and approved under the WTSA Resolution 1 procedure on 17 November 2000.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

© ITU 2001

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from ITU.

CONTENTS

	Page
1 General.....	1
2 Software tools	1
3 License and copyright.....	1
Annex A – List of software tools available	2
Annex B – ITU-T software tools General Public License.....	10
Electronic attachment: STL2000 Software Tool Library and manual	

ITU-T Recommendation G.191

Software tools for speech and audio coding standardization

1 General

In the process of generating speech and audio coding standards, the following situations often happen:

- a) in many cases, experimental results generated with different software tools may not be directly compared;
- b) software tools used by different organizations may not perfectly conform to related ITU-T Recommendations, which may delay ITU-T standardization processes;
- c) ITU-T Recommendations may leave scope for different implementations;
- d) new speech and audio coding standards are increasing in complexity, leading to non-bitexact specifications; furthermore, appropriate testing procedures to assure interoperability of different implementations are needed.

The need for a common set of tools has been recognized in past ITU-T standardization activities of speech algorithms. As a consequence, a library of portable, interworkable and reliable software routines has been established.

2 Software tools

To clarify the use of the set of software tools, arranged as a software tool library, the ITU-T makes the following recommendations:

- 1) The software tools specified in Annex A should be used as building modules of signal processing blocks to be used in the process of generation of ITU-T Recommendations, particularly those concerned with speech and audio coding algorithms.
- 2) Some of the tools shall be used in procedures for the verification of interoperability of ITU-T standards, mainly of speech and audio coding algorithms whose description is in terms of non-bitexact specifications.
- 3) The use of these modules should be made strictly in accordance with the technical instructions of their attached documentation, and should respect the following terms.

3 License and copyright

The modules in the ITU-T Software Tool Library (STL) are free software; they can be redistributed and/or modified under the terms of the "ITU-T software tools General Public License" of Annex B, as published by the ITU-T; this applies to any of the versions of the modules in the STL.

The STL has been carefully tested and it is believed that both the modules and the example programs on their usage conform to their description documents. Nevertheless, the ITU-T STL is provided "as is", in the hope that it will be useful, but without any warranty.

The STL is intended to help the scientific community to achieve new standards in telecommunications more efficiently, and for such must not be sold, entirely or in parts. The original developers, except where otherwise noted, retain ownership of their copyright, and allow their use under the terms and conditions of the "ITU-T software tools General Public License".

ANNEX A

List of software tools available

This annex contains a list with a short description of the software tools available in the ITU-T Software Tool Library. This is referred to in the associated documentation as the Software Tool Library release 2000, or STL2000. All the routines in the modules are written in C.

To obtain additional copies of the software tools listed below, as well as the associated "ITU-T Software Tool Library manual" and the above referenced application example, contact the ITU Secretariat:

ITU General Secretariat
Sales Service
Place des Nations
CH-1211 Geneva 20
Switzerland

a) *Example programs available*

Associated header file: ugstdemo.h

The following programs are examples of the use of the modules:

g711demo.c	on the use of the G.711 module.
g726demo.c	on the use of the G.726 module.
g727demo.c	on the use of the G.727 module
g722demo.c	on the use of the G.722 module.
rpedemo.c	on the use of the full-rate GSM 06.10 speech codec module.
sv56demo.c	on the use of the speech voltmeter module, and also the gain/loss routine.
eiddemo.c	on the use of the error insertion device for bit error insertion and frame erasure.
firdemo.c	on the use of the FIR (finite impulse response) high-quality low-pass and band-pass filters and of the FIR-IRS filters, associated with the rate change module.
pcmdemo.c	on the use of the G.712 (standard PCM) IIR (infinite impulse response) filters, associated with the rate change module.
filter.c	on the use of both the IIR and the FIR filters available in the rate change module.
mnrudemo.c	on the use of the narrow-band and wideband modulated noise reference unity (P.81) module.
spdemo.c	on the use of the serialization and parallelization routines of the utility module.

NOTE – The module for the Basic Operators does not have a demo program.

b) *Rate change module with FIR (finite impulse response) routines*

Name: hqflt.c

Associated header file: hqflt.h

Functions included:

<code>delta_sm_init</code>	initialize 1:1 Δ SM weighting filter.
<code>hq_down_2_to_1_init</code>	initialize 2:1 low-pass down-sampling filter.
<code>hq_down_3_to_1_init</code>	initialize 3:1 low-pass down-sampling filter.
<code>hq_up_1_to_2_init</code>	initialize 1:2 low-pass up-sampling filter.
<code>hq_up_1_to_3_init</code>	initialize 1:3 low-pass up-sampling filter.
<code>irs_8khz_init</code>	initialize 8 kHz P.48 IRS weighting filter.
<code>irs_16khz_init</code>	initialize 16 kHz P.48 IRS weighting filter.
<code>linear_phase_pb_2_to_1_init</code>	initialize 2:1 bandpass down-sampling filter.
<code>linear_phase_pb_1_to_2_init</code>	initialize 1:2 bandpass up-sampling filter.
<code>linear_phase_pb_1_to_1_init</code>	initialize 1:1 bandpass filter.
<code>mod_irs_16khz_init</code>	initialize 16 kHz send-side modified IRS weighting filter.
<code>mod_irs_48khz_init</code>	initialize 48 kHz send-side modified IRS weighting filter.
<code>psophometric_8khz_init</code>	initialize 1:1 O.41 psophometric weighting filter.
<code>p341_16khz_init</code>	initialize 1:1 P.341 send-part weighting filter for data sampled at 16 kHz.
<code>rx_mod_irs_16khz_init</code>	initialize 16 kHz modified IRS receive-side weighting filter.
<code>rx_mod_irs_8khz_init</code>	initialize 8 kHz modified IRS receive-side weighting filter.
<code>msin_16khz_init</code>	initialize mobile station weighting filter.
<code>bp5k_16khz_init</code>	initialize 5 kHz-bandwidth filter (16 kHz sampling)
<code>hq_kernel</code>	FIR filtering function.
<code>hq_reset</code>	clear state variables.
<code>hq_free</code>	deallocate FIR-filter memory.

c) *Rate change module with IIR routines*

Name: pcmflt.c

Associated header file: pcmflt.h

Functions included:

<code>stdpcm_kernel</code>	parallel-form IIR kernel filtering routine.
<code>stdpcm_16khz_init</code>	initialization of a parallel-form IIR standard PCM-filter for input and output data at 16 kHz.
<code>stdpcm_1_to_2_init</code>	as "stdpcm_16khz_init()", but needs input with sampling frequency of 8 kHz and returns data at 16 kHz.
<code>stdpcm_2_to_1_init</code>	as "stdpcm_16khz_init()", but needs input with sampling frequency of 16 kHz and returns data at 8 kHz.

<code>stdpcm_reset</code>	clear state variables (needed only if another signal should be processed with the same filter) for a parallel-form structure.
<code>stdpcm_free</code>	deallocate filter memory for a parallel-form state variable structure.
<code>cascade_iir_kernel</code>	cascade-form IIR filtering routine.
<code>iir_G712_8khz_init</code>	initialization of a cascade-form IIR standard PCM filter for data sampled at 8 kHz.
<code>iir_irs_8khz_init</code>	initialization of a cascade-form IIR P.48 IRS filter for data sampled at 8 kHz.
<code>iir_casc_lp_3_to_1_init</code>	initialization of a cascade-form IIR low-pass filter for asynchronization filtering of data and downsampling by a factor of 3:1.
<code>iir_casc_lp_1_to_3_init</code>	initialization of a cascade-form IIR low-pass filter for asynchronization filtering of data and upsampling by a factor of 3:1.
<code>cascade_iir_reset</code>	clear state variables (needed only if another signal should be processed with the same filter) for a cascade-form structure.
<code>cascade_iir_free</code>	deallocate filter memory for a cascade-form state variable structure.

d) *Error insertion module*

Name: `eid.c`

Associated header file: `eid.h`

Functions included:

<code>open_eid</code>	initializes the error pattern generator (for single-bit errors, burst bit-errors, or single frame erasures).
<code>open_burst_eid</code>	initializes the burst frame erasure pattern generator.
<code>reset_burst_eid</code>	reinitializes the burst frame erasure pattern generator.
<code>BER_generator</code>	generates a bit error sequence with properties defined by "open_eid".
<code>FER_generator_random</code>	generates a random frame erasure sequence with properties, defined by "open_eid".
<code>FER_generator_burst</code>	generates a burst frame erasure sequence with properties, defined by "open_burst_eid".
<code>BER_insertion</code>	modifies the input data bits according to the error pattern, stored in a buffer.
<code>FER_module</code>	frame erasure module.
<code>close_eid</code>	frees memory allocated to the EID state variable buffer.

e) *G.711 module*

Name: g711.c

Associated header file: g711.h

Functions included:

alaw_compress	compands 1 vector of linear PCM samples to A-law; uses 13 Most Significant Bits (MSBs) from input and 8 Least Significant Bits (LSBs) on output.
alaw_expand	expands 1 vector of A-law samples to linear PCM; uses 8 LSBs from input and 13 MSBs on output.
ulaw_compress	compands 1 vector of linear PCM samples to μ -law; uses 14 MSBs from input and 8 LSBs on output.
ulaw_expand	expands 1 vector of μ -law samples to linear PCM; uses 8 LSBs from input and 14 MSBs on output.

f) *G.726 module*

Name: g726.c

Associated header file: g726.h

Functions included:

G726_encode	G.726 encoder at 40, 32, 24 and 16 kbit/s.
G726_decode	G.726 decoder at 40, 32, 24 and 16 kbit/s.

g) *Modulated noise reference unit module*

Name: mnru.c

Associated header file: mnru.h

Functions included:

MNRU_process	module for addition of modulated noise to a vector of samples, according to ITU-T P.810, for both the narrow-band and the wideband models.
--------------	--

h) *Speech voltmeter module*

Name: sv-p56.c

Associated header file: sv-p56.h

Functions included:

init_speech_voltmeter	initializes a speech voltmeter state variable.
speech_voltmeter	measurement of the active speech level of data in a buffer according to ITU-T P.56.

i) *Module with UGST utilities*

Name: ugst-utl.c

Associated header file: ugst-utl.h

Functions included:

scale	gain/loss insertion algorithm.
sh2fl_16bit	conversion of two's complement, 16-bit integer to floating point.
sh2fl_15bit	conversion of two's complement, 15-bit integer to floating point.
sh2fl_14bit	conversion of two's complement, 14-bit integer to floating point.
sh2fl_13bit	conversion of two's complement, 13-bit integer to floating point.
sh2fl_12bit	conversion of two's complement, 12-bit integer to floating point.
sh2fl	generic function for conversion from integer to floating point.
sh2fl_alt	alternate (faster) implementation of sh2fl, with compulsory range conversion.
fl2sh_16bit	conversion of floating point data to two's complement, 16-bit integer.
fl2sh_15bit	conversion of floating point data to two's complement, 15-bit integer.
fl2sh_14bit	conversion of floating point data to two's complement, 14-bit integer.
fl2sh_13bit	conversion of floating point data to two's complement, 13-bit integer.
fl2sh_12bit	conversion of floating point data to two's complement, 12-bit integer.
fl2sh	generic function for conversion from floating point to integer.
serialize_left_justified	serialization for left-justified data.
serialize_right_justified	serialization for right-justified data.
parallelize_left_justified	parallelization for left-justified data.
parallelize_right_justified	parallelization for right-justified data.

j) *G.722 module*

Name: g722.c

Associated header file: g722.h

Functions included:

G722_encode	G.722 wideband speech encoder at 64 kbit/s.
G722_decode	G.722 wideband speech decoder at 64, 56 and 48 kbit/s.
g722_reset_encoder	initialization of the G.722 encoder state variable.
g722_reset_decoder	initialization of the G.722 decoder state variable.

k) *RPE-LTP module*

Name: rpeltp.c

Associated header file: rpeltp.h

Functions included:

rpeltp_encode	GSM 06.10 full-rate RPE-LTP speech encoder at 13 kbit/s.
rpeltp_decode	GSM 06.10 full-rate RPE-LTP speech decoder at 13 kbit/s.
rpeltp_init	initialize memory for the RPE-LTP state variables.
rpeltp_delete	release memory previously allocated for the RPE-LTP state variables.

l) *G.727 module*

Name: g727.c

Associated header file: g727.h

Functions included:

G727_encode	G.727 encoder at 40, 32, 24 and 16 kbit/s.
G727_decode	G.727 decoder at 40, 32, 24 and 16 kbit/s.

m) *Basic Operators*

Name: basop32.c

Associated header file: basop32.h

Functions included:

add	performs the addition of two 16-bit variables with overflow control and saturation; the 16-bit result is set at +32767 when overflow occurs or at -32768 when underflow occurs.
sub	performs the subtraction of two 16-bit variables with overflow control and saturation; the 16-bit result is set at +32767 when overflow occurs or at -32768 when underflow occurs.
abs_s	absolute value of a 16-bit variable; NB: abs_s(-32768) = 32767.
shl	arithmetically shift the 16-bit input left the specified number of bits. Zero fill the shifted LSB of the result. If negative, arithmetically shift right with sign extension. Saturate the result in case of underflows or overflows.

shr	arithmetically shift the 16-bit input right the specified number of bits with sign extension. If negative, arithmetically shift left by and zero fill the LSB shifted of the result.
extract_h	return the 16 MSB of the 32-bit argument.
extract_l	return the 16 LSB of the 32-bit argument.
mult	performs the multiplication of two 16-bit variables and gives a scaled 16-bit result.
L_mult	performs the 32-bit result of the multiplication of two 16-bit variables with one shift left.
negate	negate the 16-bit argument with saturation, saturate in the case when input is -32768 .
round	round the lower 16 bits of the 32-bit input number into the MS 16 bits with saturation. Shift the resulting bits right by 16 and return the 16-bit.
L_mac	multiply two 16-bit variables and shift the result left by 1. Add the 32-bit result to another 32-bit argument with saturation, returning a 32-bit result.
L_msu	multiply two 16-bit variables and shift the result left by 1. Subtract the 32-bit result from another 32-bit argument with saturation, return a 32-bit result.
L_macNs	multiply two 16-bit variables and shift the result left by 1. Add the 32-bit result to another 32-bit argument without saturation, return a 32-bit result. Generates carry and overflow values.
L_msuNs	multiply two 16-bit variables and shift the result left by 1. Subtract the 32-bit result from another 32-bit argument without saturation, return a 32-bit result. Generates carry and overflow values.
L_add	32-bit addition of the two 32-bit variables with overflow control and saturation; the result is set at $+2147483647$ when overflow occurs or at -2147483648 when underflow occurs.
L_sub	32-bit subtraction of the two 32-bit variables with overflow control and saturation; the result is set at $+2147483647$ when overflow occurs or at -2147483648 when underflow occurs.
L_add_c	performs the 32-bit addition with carry. No saturation. Generates carry and overflow values. The carry and overflow values are binary variables which can be tested and assigned values.
L_sub_c	performs the 32-bit subtraction with carry (borrow). Generates carry (borrow) and overflow values. No saturation. The carry and overflow values are binary variables which can be tested and assigned values.
L_negate	negate the 32-bit argument with saturation, saturate in the case where input is -2147483648 .
L_shl	arithmetically shift the 32-bit input left by the specified number of bits. Zero fill the LSB shifted of the result. If negative, arithmetically shift right with sign extension. Saturate the result in case of underflows or overflows.

<code>L_shr</code>	arithmetically shift the 32-bit input right by the specified number of bits with sign extension. If the number is negative, arithmetically shift left and zero fill the shifted LSBs of the result. Saturate the result in case of underflows or overflows.
<code>mult_r</code>	same as <code>mult</code> but with rounding.
<code>shr_r</code>	same as <code>shr</code> but with rounding. Saturate the result in case of underflows or overflows.
<code>shift_r</code>	same as <code>shl</code> but with rounding. Saturate the result in case of underflows or overflows.
<code>mac_r</code>	multiply two 16-bit variables and shift the result left by 1. Add the 32-bit result to a 32-bit variable with saturation. Round the LS 16 bits of the result into the MS 16 bits with saturation and shift the result right by 16. Return a 16-bit result.
<code>msu_r</code>	multiply two 16-bit values and subtract the 32-bit result from a 32-bit variable with saturation. Round the LS 16 bits of the result into the MS 16 bits with saturation and shift the result right by 16. Returns a 16-bit result.
<code>L_deposit_h</code>	deposit the 16-bit <code>var1</code> into the 16-bit MS bit of the 32-bit output. The 16 LS bits of the output are zeroed.
<code>L_deposit_l</code>	deposit the 16-bit <code>var1</code> into the 16-bit LS bit of the 32-bit output. The 16 MS bits of the output are sign extended.
<code>L_shr_r</code>	same as <code>L_shr</code> but with rounding. Saturate the result in case of underflows or overflows.
<code>L_shift_r</code>	same as <code>L_shl</code> but with rounding. Saturate the result in case of underflows or overflows.
<code>L_abs</code>	absolute value of a 32-bit variable.
<code>L_sat</code>	control overflow in 32-bit variables.
<code>norm_s</code>	produces the number of left shifts needed to normalise a positive 16-bit variable with minimum of 16384 and maximum 32767, or negative value on the interval with minimum of -32768 and maximum of -16384.
<code>div_s</code>	produces fractional integer division of two positive 16-bit values.
<code>norm_l</code>	produces the number of left shifts needed to normalise a 32-bit variable for positive values on the interval with minimum of 1073741824 and maximum 2147483647, and for negative values on the interval with minimum of -2147483648 and maximum of -1073741824
<code>L_mls</code>	performs a multiplication of a 32-bit variable by a 16-bit variable, returning a 32-bit value.
<code>div_l</code>	produces a result which is the fractional integer division of a positive 32-bit value by a positive 16-bit value. The result is positive (leading bit equal to 0) and truncated to 16 bits.
<code>i_mult</code>	multiply two 16-bit words returning a 16-bit word with overflow control.

<code>L_mult0</code>	performs the 32-bit result of the multiplication of two 16-bit variables without one shift left.
<code>L_mac0</code>	multiply two 16-bit variables without left shift by 1 of the result. Add the 32-bit result to another 32-bit argument with saturation, returning a 32-bit result.
<code>L_msu0</code>	multiply two 16-bit variables without left shift by 1 of the result. Subtract the 32-bit result from another 32-bit argument with saturation, return a 32-bit result.

ANNEX B

ITU-T software tools General Public License

Terms and conditions

B.1 This License Agreement applies to any module or other work related to the ITU-T Software Tool Library, and developed by the User's Group on Software Tools. The term "Module", below, refers to any such module or work, and a "work based on the Module" means either the Module or any work containing the Module or a portion of it, either verbatim or with modifications. Each licensee is addressed as "you".

B.2 You may copy and distribute verbatim copies of the Module's source code as you receive it, in any medium, provided that you:

- conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty;
- keep intact all the notices that refer to this General Public License and to the absence of any warranty; and
- give any other recipients of the Module a copy of this General Public License along with the Module.

You may charge a fee for the physical act of transferring a copy.

B.3 You may modify your copy or copies of the Module or any portion of it, and copy and distribute such modifications under the terms of B.1, provided that you also do the following:

- Cause the modified files to carry prominent notices stating that you changed the files and the date of any change; and
- Cause the whole of any work that you distribute or publish, that in whole or in part contains the Module or any part thereof, either with or without modifications, to be licensed at no charge to all third parties under the terms of this General Public License (except that you may choose to grant warranty protection to some or all third parties, at your option).
- If the modified module normally reads commands interactively when run, you must cause it, when started running for such interactive use in the simplest and most usual way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the module under these conditions, and telling the user how to view a copy of this General Public License.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

Mere aggregation of another independent work with the Module (or its derivative) on a volume of a storage or distribution medium does not bring the other work under the scope of these terms.

B.4 You may copy and distribute the Module (or a portion or derivative of it, under B.2) in object code or executable form under the terms of B.1 and B.2, provided that you also do one of the following:

- accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of B.1 and B.2; or
- accompany it with a written offer, valid for at least three years, to give any third party free (except for a nominal charge for the cost of distribution) a complete machine-readable copy of the corresponding source code, to be distributed under the terms of B.1 and B.2; or
- accompany it with the information you received as to where the corresponding source code may be obtained. (This alternative is allowed only for non-commercial distribution and only if you received the module in object code or executable form alone.)

Source code for a work means the preferred form of the work for making modifications to it. For an executable file, complete source code means all the source code for all modules it contains; but, as a special exception, it need not include source code for modules which are standard libraries that accompany the operating system on which the executable file runs, or for standard header files or definitions files that accompany that operating system.

B.5 You may not copy, modify, sublicense, distribute or transfer the Module except as expressly provided under this General Public License. Any attempt otherwise to copy, modify, sublicense, distribute or transfer the Module is void, and will automatically terminate your rights to use the Module under this License. However, parties who have received copies, or rights to use copies, from you under this General Public License will not have their licenses terminated so long as such parties remain in full compliance.

B.6 By copying, distributing or modifying the Module (or any work based on the Module) you indicate your acceptance of this license to do so, and all its terms and conditions.

B.7 Each time you redistribute the Module (or any work based on the Module), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Module subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein.

B.8 The ITU-T may publish revised and/or new versions of this General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Module specifies a version number of the license which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the ITU-T. If the Module does not specify a version number of the license, you may choose any version ever published by the ITU-T.

B.9 If you wish to incorporate parts of the Module into other free modules whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the ITU-T, write to the ITU-T Secretariat; exceptions may be made for this. This decision will be guided by the two goals of preserving the free status of all derivatives of this free software and of promoting the sharing and reuse of software generally.

B.10 Because the Module is licensed free of charge, there is no warranty for the Module, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holders and/or other parties provide the Module "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the Module is with you. Should the Module prove defective, you assume the cost of all necessary servicing, repair or correction.

B.11 In no event unless required by applicable law or agreed to in writing will any copyright holder, or any other party who may modify and/or redistribute the Module as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the Module (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the Module to operate with any other modules), even if such holder or other party has been advised of the possibility of such damages.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series B	Means of expression: definitions, symbols, classification
Series C	General telecommunication statistics
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	TMN and network maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks and open system communications
Series Y	Global information infrastructure and Internet protocol aspects
Series Z	Languages and general software aspects for telecommunication systems

20702