TINX 9.1 reference card

Application structure, Basic Temporal Logic syntax and user interface features
Version 1.0.0

Application structure

Modules

Module	Description
tinx	Temporal inference network executor, single core version of the inference engine
tinx_mt	Temporal inference network executor, multiple core version of the inference engine
ting	Temporal inference network generator, the Basic Temporal Logic compiler
gtinxsh	Graphical TINX shell, the interactive development environment

Files

File	Description
*.btl	Basic Temporal Logic source files, editable text
*.tin	Temporal inference network object files, text generated by module “ting”
*.evl	Event list files, text generated by module “ting” to describe the initial conditions of execution and by module “tinx” as an inference process log
*.io	Input and output signal files, text containing binary symbols or 8-bit characters and generated by modules “tinx” e “gtinxsh”
*.sym	Symbol table files, text generated by module “ting”
.gtinxshrc	Configuration file of the graphical shell, binary

Paths

Path	Description
/usr/bin/	Executable modules
/usr/share/doc/tinx/	Documentation in PDF and ASCII formats
/usr/share/tinx/examples/	Examples of binary dynamical system specifications in Basic Temporal Logic source files and temporal inference network object files
/usr/share/tinx/sources/	C language and metacompiler complete source code

Basic Temporal Logic

General instruction syntax

Syntax	Description
P1; *P2*; … *Pn*;	Set of temporal logic clauses or declarations *P1*, *P2*, … *Pn*
P(*K*)	Parametric vector clause *P* with index *K*
P(*H*, *K*)	Parametric matrix clause *P* with indexes *H* and *K*
// *Text*	C++ language style comment
/* *Text* */	C language style comment

Logic operators

Syntax	Description
~ *P*	Not clause *P*
P & *Q*	Clause *P* and clause *Q*
P | *Q*	Clause *P* or clause *Q*
P --> Q	Clause *P* implies clause *Q*
P <-- *Q*	Clause *P* is implied by clause *Q*
P == *Q*	Clause *P* double implies clause *Q*
forall(*P*(*K*), *K* on *N*)	For all parametric clauses *P*(*K*) between 0 and *N* – 1
forall(*P*(*K*), *K* in *A* : *B*)	For all parametric clauses *P*(*K*) on a range from *A* to *B*
exists(*P*(*K*), *K* on *N*)	Exists a parametric clause *P*(*K*) between 0 and *N* – 1
exists(*P*(*K*), *K* in *A* : *B*)	Exists a parametric clause *P*(*K*) on a range from *A* to *B*
unique(*P*(*K*), *K* on *N*)	Exists an unique parametric clause *P*(*K*) between 0 and *N* – 1
unique(*P*(*K*), *K* in *A* : *B*)	Exists an unique parametric clause *P*(*K*) on a range from *A* to *B*
one(*P*(*K*), *K* is *A*, *K* on *N*) one(*P*(*K*), not *K* is *A*, *K* on *N*)	Asserts only one parametric clause *P*(*A*) while negating the other clauses *P*(*K*) between 0 and *N* – 1 or otherwise
one(*P*(*K*), *K* in *A* : *B*, *K* on *N*) one(*P*(*K*), not *K* in *A* : *B*, *K* on *N*)	Asserts a subrange from *A* to *B* of parametric clauses *P*(*K*) while negating the other clauses *P*(*K*) between 0 and *N* – 1 or otherwise
one(*P*(*K*), *K* is *A*, *K* in *C* : *D*) one(*P*(*K*), not *K* is *A*, *K* in *C* : *D*)	Asserts only one parametric clause *P*(*A*) while negating the other clauses *P*(*K*) on a range from *C* to *D* or otherwise
one(*P*(*K*), *K* in *A* : *B*, *K* in *C* : *D*) one(*P*(*K*), not *K* in *A* : *B*, *K* in *C* : *D*)	Asserts a subrange from *A* to *B* of parametric clauses *P*(*K*) while negating the other clauses *P*(*K*) on a range from *C* to *D* or otherwise

Temporal operators

Syntax	Description
P @ *T*	Clause *P* at relative time *T*
P @ [*A*, *B*]	For all clauses *P* between relative time *A* and time *B*, included
P @ (*A*, *B*]	For all clauses *P* between relative time *A* + 1 and time *B*, included
P @ [*A*, *B*)	For all clauses *P* between relative time *A* and time *B* - 1, included
P @ (*A*, *B*)	For all clauses *P* between relative time *A* + 1 and time *B* - 1, included
P ? [*A*, *B*]	Exists a clause *P* between relative time *A* and time *B*, included
P ? (*A*, *B*]	Exists a clause *P* between relative time *A* + 1 and time *B*, included
P ? [*A*, *B*)	Exists a clause *P* between relative time *A* and time *B* - 1, included
P ? (*A*, *B*)	Exists a clause *P* between relative time *A* + 1 and time *B* - 1, included
since(*P*, *Q*)	Since clause *P* is true until clause *Q* keeps true
until(*P*, *Q*)	Until clause *P* is true since clause *Q* keeps true

Iterations and selections

Syntax	Description
iter(*K* on *N*) *P*(*K*)	Repeats a parametric clause or declaration *P*(*K*) between 0 and *N* – 1
iter(*K* on *N*) { *P1*(*K*); *P2*(*K*); … *Pn*(*K*); }	Repeats a set of parametric clauses or declarations *P1*(*K*), *P2*(*K*), … *Pn*(*K*) between 0 and *N* – 1
iter(*K* in *A* : *B*) *P*(*K*)	Repeats a parametric clause or declaration *P*(*K*) on a range from *A* to *B*
iter(*K* in *A* : *B*) { *P1*(*K*); *P2*(*K*); … *Pn*(*K*); }	Repeats a set of parametric clauses or declarations *P1*(*K*), *P2*(*K*), … *Pn*(*K*) on a range from *A* to *B*
when(*K* is *A*) *P*(*K*) when(not *K* is *A*) *P*(*K*) when(*K* is *A*) *P*(*K*) else *Q*(*K*) when(not *K* is *A*) *P*(*K*) else *Q*(*K*)	Selects or excludes only one parametric clause or declaration *P*(*A*) within a loop, eventually while substituting it with clause or declaration *Q*(*K*) in the other iterations
when(*K* is *A*) { *P1*(*K*); *P2*(*K*); … *Pn*(*K*); } when(not *K* is *A*) { *P1*(*K*); *P2*(*K*); … *Pn*(*K*); } when(*K* is *A*) { *P1*(*K*); *P2*(*K*); … *Pn*(*K*); } else { *Q1*(*K*); *Q2*(*K*); … *Qn*(*K*); } when(not *K* is *A*) { *P1*(*K*); *P2*(*K*); … *Pn*(*K*); } else { *Q1*(*K*); *Q2*(*K*); … *Qn*(*K*); }	Selects or excludes only one set of parametric clauses or declarations *P1*(*A*), *P2*(*A*), … *Pn*(*A*) within a loop, eventually while substituting them with clauses or declarations *Q1*(*K*), *Q2*(*K*), … *Qn*(*K*) in the other iterations

Syntax	Description
when(*K* in *A* : *B*) *P*(*K*) when(not *K* in *A* : *B*) *P*(*K*) when(*K* in *A* : *B*) *P*(*K*) else *Q*(*K*) when(not *K* in *A* : *B*) *P*(*K*) else *Q*(*K*)	Selects or excludes a subrange from *A* to *B* of parametric clauses or declarations *P*(*K*) within a loop, eventually while substituting them with clauses or declarations *Q*(*K*) in the other iterations
when(*K* in *A* : *B*) { *P1*(*K*); *P2*(*K*); … *Pn*(*K*); } when(not *K* in *A* : *B*) { *P1*(*K*); *P2*(*K*); … *Pn*(*K*); } when(*K* in *A* : *B*) { *P1*(*K*); *P2*(*K*); … *Pn*(*K*); } else { *Q1*(*K*); *Q2*(*K*); … *Qn*(*K*); } when(not *K* in *A* : *B*) { *P1*(*K*); *P2*(*K*); … *Pn*(*K*); } else { *Q1*(*K*); *Q2*(*K*); … *Qn*(*K*); }	Selects or excludes a subrange from *A* to *B* of sets of parametric clauses or declarations *P1*(*K*), *P2*(*K*), … *Pn*(*K*) within a loop, eventually while substituting them with clauses or declarations *Q1*(*K*), *Q2*(*K*), … *Qn*(*K*) in the other iterations

Declarations

Syntax	Description
input *P1*, *P2*, … *Pn*	Declares a set of input signals *P1*, *P2*, … *Pn* corresponding to each clause
input [*O1*, *O2*, *O3*, *O4*] *P1*, *P2*, … *Pn*	Declares a set of input signals *P1*, *P2*, … *Pn* corresponding to each clause with specific options *O1*, *O2*, *O3*, *O4*
output *P1*, *P2*, … *Pn*	Declares a set of output signals *P1*, *P2*, … *Pn* corresponding to each clause
output [*O1*, *O2*, *O3*, *O4*] *P1*, *P2*, … *Pn*	Declares a set of output signals *P1*, *P2*, … *Pn* corresponding to each clause with specific options *O1*, *O2*, *O3*, *O4*
aux *P1*, *P2*, … *Pn*	Declares a set of auxiliary signals *P1*, *P2*, … *Pn* corresponding to each clause
init *P1* @ *T1*, *P2* @ *T2*, … *Pn* @ *Tn*	Declares as true the clauses *P1*, *P2*, … *Pn* respectively at absolute times *T1*, *T2*, … *Tn*
init ~ *P1* @ *T1*, ~ *P2* @ *T2*, … ~ *Pn* @ *Tn*	Declares as false the clauses *P1*, *P2*, … *Pn* respectively at absolute times *T1*, *T2*, … *Tn*
init *P1* @ [*A1*, *B1*], *P2* @ [*A2*, *B2*], … *Pn* @ [*An*, *Bn*]	Declares as true the clauses *P1*, *P2*, … *Pn* respectively between absolute time *A1* and time *B1*, *A2* and *B2*, … *An* and *Bn*, included
init ~ *P1* @ [*A1*, *B1*], ~ *P2* @ [*A2*, *B2*], … ~ *Pn* @ [*An*, *Bn*]	Declares as false the clauses *P1*, *P2*, … *Pn* respectively between absolute time *A1* and time *B1*, *A2* and *B2*, … *An* and *Bn*, included
define *K1* = *E1*, *K2* = *E2*, … *Kn* = *En*	Computes the integer expressions *E1*, *E2*, … *En* and assigns their results respectively to the variables *K1*, *K2*, … *Kn*

Other constructs

Syntax	Description
include “*F1*”, “*F2*”, … “*Fn*”	Includes in the current program the BTL source files named *F1*, *F2*, … *Fn*
code(*P*, “*S*”)	Asserts and negates properly the parametric clauses *P*(*H*, *K*) so as to represent the string *S*, with the character number in *H* and the bit number in *K*
code(*P*, *N*)	Asserts and negates properly the parametric clauses *P*(*H*, *K*) so as to represent the unsigned integer *N*, with the byte number in *H* and the bit number in *K*
#	Represents the iterator of the current loop, valid also if no variable is specified
## ### …	Represent the iterators of the outer loops, valid also if no variable is specified

Input and output options

Option	Description
any	Signal input and output channels are determined by external options (default)
ipc	Signal input and output occur always by inter process communication
file	Signal input and output occur always by shared file
binary	Input and output signals contain binary symbols (default)
packed	Input and output signals contain strings of 8-bit characters
false	The binary signal or packed signal bit is false as default
true	The binary signal or packed signal bit is true as default
unknown	The binary signal or packed signal bit is unknown as default (default)
raw	The binary signal or packed signal bit is always either waited for acquisition for inputs or produced for outputs (default)
filter	The binary signal or packed signal bit is either acquired within a time unit or given a default value for inputs and it is produced only if determined of a value different from the default for outputs
omit	The binary signal or packed signal bit is either acquired within a time unit or given a default value for inputs and it is produced only if determined of a value different from both unknown and the default for outputs

Default binary input and output symbols

Symbol	Description
0	False
1	True
?	Unknown
.	The signal reached its end and can be dropped as a source of information, when all the input signals are dropped the inference and outputs proceed anyway until possible
Escape	An immediate termination of the inference process and outputs is requested (the same result can be obtained by sending an interrupt signal or Ctrl-C to the inference engine)

Mathematical operators

Syntax	Description
A + *B*	Sum of number *A* and number *B*
A - *B*	Difference between number *A* and number *B*
A * *B*	Product of number *A* for number *B*
A / *B*	Quotient of number *A* for number *B*
A % *B*	Reminder of the division between number *A* and number *B*
A ^ *B*	Power of number *A* to number *B*

String escape sequences

Sequence	Description
\\	Backslash
\”	Double quote
\a	Alert
\b	Backspace
\f	Form feed
\n	New line
\r	Carriage return
\t	Horizontal tab
\v	Vertical tab
\x*N*	Character with hexadecimal code *N*

User interface

Main controls

Name	Switch	Description
Temporal logic source file name (text field)	ting	Name without extension of the BTL source file, the side icon opens the file chooser dialog
Load initial conditions (check box)	tinx -i	The initial conditions of execution are evaluated, this is necessary for all systems without inputs or with unreachable state (see “init” construct)
Edit… (button)		Opens the default text editor on the current BTL source file (see “Default text editor” field)
Sampling time (numeric field)	tinx -t	Sampling time in seconds and fractions or zero for fast execution mode
Horizon length (numeric field)	tinx -z	Overall duration of execution expressed in number of samples to process or zero for an unlimited temporal horizon (default option)
Verify inference soundness (check box)	tinx -v	The production of a contradiction at run time interrupts the execution, instead of proceeding anyway
Causal inference only (check box)	tinx -c	The inference proceeds from past or present to present or future only, some system will not work with this performance option enabled
Use symbol table (check box)	ting -x tinx -x	The inference process trace or log includes also BTL clauses
IPC input and output (radio button)	(default)	Input and output of signals unconstrained to a specific method are performed by inter process communication (see “input” and “output” constructs and their options)
File input and output (radio button)	tinx -f	Input and output of signals unconstrained to a specific method are performed by shared files (see “input” and “output” constructs and their options)
Quiet mode (radio button)	tinx -q	No input or output are performed, all systems with inputs will not work with this performance option enabled
Trace inference (check box)	tinx -d	A trace at run time of the inference process is displayed in the message frame, this may break the real time (see “Use symbol table” option)
Log inference to file (check box)	tinx -l	A log at run time of the inference process is produced as a text file (see “Use symbol table” option)
Display auxiliary signals (check box)	ting -b	Also the signals declared as auxiliary are displayed in the history frame as outputs with default options (see “aux” construct)

Name	Switch	Description
Input truth probability (slider)		Truth probability of binary symbols in all randomly generated input signals, to be set to one if different values are chosen for each signal (see “Probabilities…” window)
Time correction (slider)		Difference between clocks of the graphical shell and the inference engine, experimental
Generate network (button)		Compiles the BTL source file to generate a temporal inference network and its execution initial conditions with current options
Execute network (button)		Activates the inference engine and executes the temporal inference network with current options
Stop execution (button)		Interrupts the execution of the temporal inference network
Help… (button)		Opens the help window (see “Default help viewer” field)
Configure… (button)		Opens the configuration window
Exit (button)		Exits from the graphical shell and halts the inference engine if running

Display format

Drawing	Description
Red sample sequence	Binary input signal history
Green sample sequence	Binary output signal history
Orange sample sequence	8-bit character input signal history
Yellow sample sequence	8-bit character output signal history
Turquoise sample sequence	Auxiliary or internal signal history (with “Display auxiliary signals” or “Display internal signals” options enabled)
Small rectangle	False bit
Big rectangle	True bit
Dot	Unknown bit (with “Display unknowns as dots” option enabled)
Yellow circle	Loss of phase in output signal and real time broken
Yellow ring	Binary output signal of unknown value in phase
Elapsed time	Partial duration of execution expressed in number of samples processed and number of corresponding seconds, compared to measured time expressed in seconds
Red exclamation point	Warning symbol asserting it is necessary to regenerate the temporal inference network so that the requested options can take place

Configuration controls

Name	Switch	Description
Network object file name (text field)	ting -o tinx	Name without extension of the temporal inference network object file, the side icon opens the file chooser dialog
Initial conditions file name (text field)	ting -I tinx -I	Name without extension of the execution initial conditions file, the side icon opens the file chooser dialog (see “Load initial conditions” option)
Log file name (text field)	tinx -L	Name without extension of the inference process log file, the side icon opens the file chooser dialog (see “Log inference to file” option)
Symbol table file name (text field)	ting -X tinx -X	Name without extension of the symbol table file, the side icon opens the file chooser dialog (see “Use symbol table” option)
Module file path (text field)	ting -P	Name of the module file path, the side icon opens the file chooser dialog (see “include” construct)
Default text editor (text field)		Name of the executable file of the default text editor for BTL source files (see “Edit…” command)
Default help viewer (text field)		Name of the executable file of the default PDF viewer for help files (see “Help…” command)
IPC prefix (text field)	tinx -e	Prefix for inter process communication port names, it has to start with a “/” (see “IPC input and output” option and “input” and “output” constructs)
I/O file path (text field)	tinx -p	Path for shared file names used for input and output signals (see “File input and output” option and “input” and “output” constructs)
False char (text field)	tinx -a	Symbol used to represent the false in binary mode, it has to be the first character of the command line argument
True char (text field)	tinx -a	Symbol used to represent the true in binary mode, it has to be the second character of the command line argument if present
Unknown char (text field)	tinx -a	Symbol used to represent the unknown in binary mode, it has to be the third character of the command line argument if present
End char (text field)	tinx -a	Symbol used to represent the end of input or output signals in binary mode, it has to be the fourth character of the command line argument if present
External inputs (check box)		The inference engine gathers all its input signals from external programs or files and no random input signal is generated or displayed by the graphical shell, disabling this option may overwrite predetermined external files

Name	Switch	Description
External outputs (check box)		The inference engine provides all its output signals to external programs or files and no output signal is displayed by the graphical shell
System V IPC (check box)	tinx -V	Inter process communication follows the System V protocol instead of the POSIX protocol, the uniqueness of the port identifiers is no more granted
Ignore IPC errors (check box)	tinx -y	Inter process communication errors are ignored instead of causing the end of input or output signals
Display internal signals (check box)	ting -B	Also the internal signals used in the decomposition of the temporal operators are displayed in the history frame as outputs with default options
Display unknowns as dots (check box)		Unknown logical values are displayed as dots in the history frame
Number of processes (numeric field)	tinx_mt -n	If equal to one the single core version of the inference engine is used, if more than one such a number of cores are allocated to the inference process by the multiple core version plus one core for input and output
Memory size logarithm (numeric field)	tinx -r	Two raised to this number is proportional to the memory allocated for the inference process, it should be more than four times the maximum time present in the arguments of temporal interval operators (see “@” and “?” constructs)
History window columns (numeric field)		Number of samples displayed in the history frame
History window max rows (numeric field)		Maximum number of signals displayed in the history frame
Optimize joints in intervals (check box)	ting -w	The number of joints of the temporal inference network produced by the interval operators is optimized (see “@” and “?” constructs)
Optimize joints in recursions (check box)	ting -W	The number of joints of the temporal inference network produced by the recursive operators is optimized, this may cause problems with some initial conditions (see “since” and “until” constructs)
Optimize delays (check box)	ting -u	The number of delays of the temporal inference network is optimized, this may cause problems with the selected memory size
Generate constant outputs (check box)	ting -k	Constant output signals are generated anyway and not removed by the optimizer
Hard real time (check box)	tinx -s	A fair scheduling is requested to the operating system, *this option is available at root access only*

Name	Switch	Description
Wait running (check box)	tinx -S	The inference engine busy waits, this will increase the system overhead but decrease the inference process latency
Display full signal names (check box)		Signal names are displayed with port identifiers or path and extension in the history frame
Save configuration (button)		Saves the graphical shell configuration in the current directory
Probabilities… (button)		Opens a window containing the truth probabilities of binary symbols in each single randomly generated input signal, so different values can be chosen (see “Input truth probability” slider, which has to be set to one for this option)
External signals… (button)		Opens a window containing the list of input and output signals which have to be managed by external programs and not by the graphical shell (see “External inputs” and “External outputs” options)
Clear configuration (button)		Deletes the graphical shell configuration from current directory
Clear (button)		Resets to one truth probabilities of binary symbols in all randomly generated input
	tinx_mt -D	Displays debug information of the multiple core version only
	tinx -g	Sets the origin of the temporal reference system, equal to the current time if omitted

Inference engine trace and log file format

Record	Description
(*V*, *W*) # *K* @ *I* (*V*, *W*) # *K* @ *I*: *P* --> *Q*	In the log file, the edge from vertex *V* to vertex *W* with index *K* representing clause *Q* at absolute time *I*, clause which has been deduced from clause *P* at some time, is selected and processed for further inference
> *T*: (*V*, *W*) # *K* @ *I* > *T*: (*V*, *W*) # *K* @ *I*: *P* --> *Q* #*N* > *T*: (*V*, *W*) # *K* @ *I* #*N* > *T*: (*V*, *W*) # *K* @ *I*: *P* --> *Q*	In the execution trace, described processing is performed at external absolute time *T* by the core number *N* for the multiple core version
* #*N* *	In the execution trace, no further inference is possible until new inputs and this is noted by the core number *N* for the multiple core version
| #*N* |	In the execution trace, no further inference will be done until external time advances and this is noted by the core number *N* for the multiple core version

Temporal inference network file format

Record	Description
V0: G ; *V1*, *V2*, *V3* *V0*: J ; *V1*, *V2*, *V3* *V0*: D *T* ; *V1*, *V2*	Vertex *V0* of class gate (“G”), joint (“J”) or delay (“D”) of degree *T* is connected to parent vertex *V1* and to sons vertexes *V2* and possibly *V3*
! *P* (*V*, *W*) # *K* / *O1*, *O2’*, *O2”*, *O3*, *O4* ? *P* (*V*, *W*) # *K* / *O1*, *O2’*, *O2”*, *O3*, *O4* . *P* (*V*, *W*) # *K* / *O1*, *O2’*, *O2”*, *O3*, *O4*	Input signal (“!”), output signal (“?”) or auxiliary signal (“.”) corresponding to clause *P* is represented by the edge from vertex *V* to vertex *W* with index *K* and options *O1*, *O2*, *O3* and *O4* (*O2’* is the packed signal number if not zero and *O2”* the packed signal bit for that number)