FBLIST Example

A Much Better Idea : Logical : Unique

AmbiLogique

Electronic Controllers

The FBLIST example is a multi-page AmbiLogique diagram which lists all of the available function blocks in the CPDA-01 library.

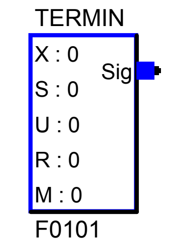
Each function is illustrated, and its description and pin functions are given alongside.

The functions are grouped by page in exactly the same way as they are grouped in the Insert Function dialog in the AmbiLogique_PLC software.

AmbiLogique Ltd 1812 Opunake Road, RD29, Hawera 4679, New Zealand

+64 6 764 6567 ph

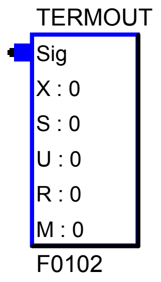
www.ambilogique.com   sales@ambilogique.com



Type 4 - TERMIN

This function block is used for all input terminals.
When it has been inserted into the diagram, right-clicking on it will bring up
the edit menu, allowing you to select 'Properties.'
This in turn brings up a special dialogue which permits you to fill
in the constants for the terminal.

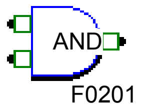
boX is the AmbiLogique backplane selector. For non-network controllers, it is 0.
Slot is the slot into which the processor or expansion module is plugged.
Processors plug into slot 0.
sUbslot is used with backplane extenders, which allow you to plug extra
backplanes into any numbered slot on the primary backplane.
For modules plugged into the primary backplane, sUbslot is 0.
Some processor module facilities are accessible to the diagram via subslots other than 0.
Register and Mask values are given on the data sheet for the
Processor or Expansion module.



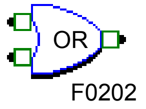
Type 5 - TERMOUT

This function block is used for all output terminals.
When it has been inserted into the diagram, right-clicking on it
will bring up the edit menu, allowing you to select 'Properties.'
This in turn brings up a special dialogue which permits you to fill
in the constants for the terminal.

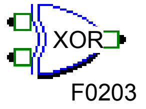
boX is the AmbiLogique backplane selector. For non-network controllers, it is 0.
Slot is the slot into which the processor or expansion module is plugged.
Processors plug into slot 0.
sUbslot is used with backplane extenders, which allow you to plug extra
backplanes into any numbered slot on the primary backplane.
For modules plugged into the primary backplane, sUbslot is 0.
Some processor module facilities are accessible to the diagram via subslots other than 0.
Register and Mask values are given on the data sheet for the
Processor or Expansion module.



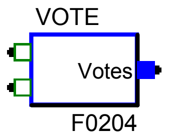
Type 6 - AND
The AND gate output is TRUE (1) if all of the inputs are TRUE (non-zero).
The output is FALSE (0) if any of the inputs is FALSE (zero).
Inputs can be added up to 15 in all.
Any input or the output can be inverted.



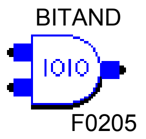
Type 7 - OR
The OR gate output is TRUE (1) if any of the inputs is TRUE (non-zero).
The output is FALSE (0) if all of the inputs are FALSE (zero).
Inputs can be added up to 15 in all.
Any input or the output can be inverted.



Type 8 - XOR
The XOR gate output is TRUE (1) if
an odd number of the inputs are TRUE (non-zero).
The output is FALSE (0) if an even number of the inputs are TRUE.
Inputs can be added up to 15 in all.
Any input or the output can be inverted.

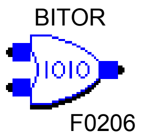


Type 9 - VOTE
The VOTE function output is the count of the inputs which are TRUE (non-zero).
Inputs can be added up to 15 in all.
Any input or the output can be inverted.



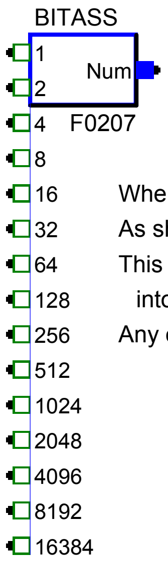
Type 10 - BITAND
The BITAND Bitwise AND function handles analogue instead of digital signals.
The input signals are aligned and the individual bits are ANDed together.

As an example, take signals of value 7 (00111) and 30 (11110).
If these were fed into an AND gate, both are non-zero (TRUE), so the result is also TRUE.
Feeding these into the BITAND gate gives a result 00110 = 6.
Inputs can be added up to 15 in all.



Type 11 - BITOR
The BITOR Bitwise OR function handles analogue instead of digital signals.
The input signals are aligned and the individual bits are ORed together.

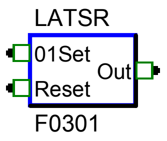
As an example, take two signals of value 7 (00111) and 25 (11001).
If these were fed into an OR gate, both are non-zero (TRUE), so the result is also TRUE.
Feeding these into the BITOR gate gives a result 11111 = 31.
Inputs can be added up to 15 in all.



Type 46 - BITASS
BITASS is a bit assembler which takes 1's on the inputs and assembles them into corresponding bits of the output Number.
This means that if the '1' input is TRUE, 1 is added to Num; if the '2' input is TRUE, 2 is added to Num; and so on.
When first added to a diagram, the BITASS block has 2 inputs.
As shown here, the block can be expanded up to 15 inputs in all.
This BITASS function block is great for combining several digital signals into a single composite signal.
Any of the inputs can be inverted.

AmbiLogique Ltd

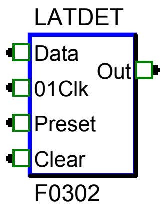
FUNCTION BLOCK LIST - 2 - LOGIC GATES



Type 14 - LATSR - Set/Reset Latch with Edge-triggered Set

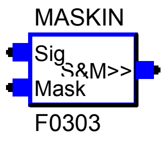
The SR (Set/Reset) latch output becomes TRUE (1) when the 01Set input changes from FALSE (0) to TRUE (non-zero), provided that Reset is FALSE.
The output remains TRUE until Reset becomes TRUE.
Reset overrides: the output cannot become TRUE if Reset is TRUE.

If Reset is TRUE and 01Set changes from FALSE to TRUE, the output remains FALSE.
If Reset then becomes FALSE with 01Set remaining TRUE, the output remains FALSE.
Either input or the output can be inverted.



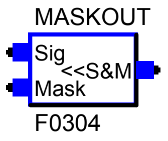
Type 15 - LATDET - Data type latch with Edge-triggered Clock

The DET (D type Edge Triggered) latch transfers the state of the Data signal
to the output when the 01Clk input changes from FALSE (0) to TRUE (non-zero),
or until provided that both Clear and Preset are FALSE.
The output holds its state until there is another FALSE to TRUE transition of 01Clk,
OR Clear or Preset become TRUE.
As soon as Clear becomes TRUE, the output is forced to FALSE.
If Clear is FALSE and Preset is TRUE, the outptut is forced to TRUE.
If Clear is TRUE and 01Clk changes from FALSE to TRUE, even though Data may be TRUE,
the output remains FALSE.
if Clear is FALSE and Preset is TRUE, the output remains TRUE.
To summarise, Clear overrides Preset and 01Clk. Preset overrides 01Clk.
Any input or the output can be inverted.



Type 12 - MASKIN

This function is used internally by AmbiLogique for extracting values from registers which have multiple values in them.
An example of this type of register is Isw0_7 in the CPDA-01 controller.
Each switch input on the controller is mapped into a single bit in this register.
The function consists of a bitwise AND (just like the BITAND function) together with a shifter which places the least significant bit in position 0.
As an example, take a signal of value 19 (10011) and a mask value of 24 (11000).
When these are ANDed together, the result is 10000 (16).
It takes 3 shifts to the right to bring the least significant bit of the mask into position 0;
so we shift our result 3 places to the right. The result is 00010 (2).
The MASKIN function clips out part of a register, then shifts it so that becomes a normal integer.



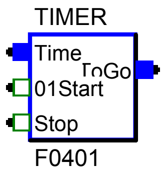
Type 13 - MASKOUT

This function is used internally by AmbiLogique for inserting values into registers which hold multiple values.
An example of this type of register is Otr0_7 in the CPDA-10 controller.

The function consists of a shifter which aligns the signal with the mask,
then carries out a bitwise AND with the mask.
As an example, take have a signal value of 19 (10011) and a mask value of 24 (11000).
The signal needs to be shifted 3 places left to align its least significant bit
with the least significant '1' in the mask.
This makes the shifted signal 10111000.
ANDing the shifted signal with the mask gives a result of 00011000 or 24.

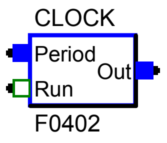
AmbiLogique Ltd

FUNCTION BLOCK LIST - 3 - LATCHES & MEMORIES
BITSHIFTERS



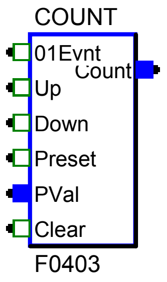
Type 16 - TIMER

This timer counts 1/16 second intervals.
The output is 0 if Stop is TRUE.
If Stop is FALSE and 01Start changes from FALSE to TRUE, the value at the Time input is transferred to the output, and the output value counts down to 0.
When the count gets to 0, counting stops.
During counting, if 01Start changes from TRUE to FALSE, counting continues without interruption.
During counting, if Stop becomes TRUE, the output goes to 0 and counting stops.
The time remaining in the timer is lost.
The output can be connected to a digital input of another function block, where it will be seen as TRUE (non-zero) while the timer is counting, and FALSE when the time has expired or has been stop
The 01Start or the Stop input can be inverted.



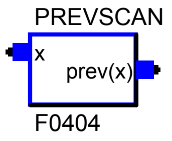
Type 17 - CLOCK - Continuous timebase (repeat waveform)

This clock function outputs a continuous train of staircase waves.
The waveform period is expressed in 1/16ths of a second.
When Run changes from FALSE to TRUE, the output goes to (Period - 1) for 1/16 second, then counts down to 0.
After 1/16 second in the 0 state, the outputs returns to (Period - 1) and counts down again.
The cycle repeats until Run goes FALSE, when the output goes to 0 immediately.
Note that the cycle is synchronised to the FALSE/TRUE transition of Run;
it does not continue in the background while Run is FALSE.
If the output of the clock is wired to a digital input, it will be TRUE for (Period - 1)/16 second, and FALSE for 1/16 second.
More complex waveforms can be generated by wiring the Clock output to a Compare function or to the 01Start input of a Timer.
The Run input can be inverted.



Type 18 - COUNT

This counter counts FALSE/TRUE transitions on the 01Evnt input.
The count increases if Up is TRUE during one of the transitions, and decreases if Down is TRUE.
If both Up and Down are TRUE during a transition, the count is unchanged.
The Preset input overrides Count, Up and Down.
Preset, when TRUE, will force the PVal (Preset Value) into the counter and therefore the output.
Preset can be overriden by Clear.
Clear overrides all other inputs and forces the counter (and the output) to 0 when it is TRUE.
The 01Evnt, Up, Down, Preset and Clear inputs can be inverted.
This counter requires that TRUE and FALSE periods on 01Evnt are at least 1/8 second long for correct
The counter can therefore cope with 4 counts/second maximum.
For faster counts, look at the built-in high-speed counters attached to inputs ISW6 and ISW7.

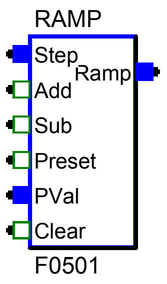


Type 1 - PREVSCAN

The PrevScan function outputs the value of the input signal 'x' on the previous scan (i.e. 1/16 second ago.)
This enables the rate of change of a signal to be calculated.
If the difference between the input and output values is multiplied by 16, we have the rate of change per second of the input signal.
This is useful for computing the D function in PD or PID control.
PREVSCAN blocks may be cascaded to form a shift register to store signals over longer periods.

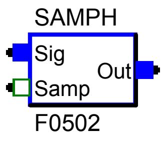
AmbiLogique Ltd

FUNCTION BLOCK LIST - 4 - TIMERS & COUNTERS



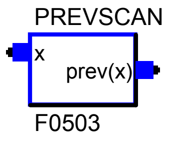
Type 19 - RAMP

This RAMP function is an analogue version of the up/down counter.
It can be used to generate soft starts in motion control, trapezoidal waveforms,
controlled rates of change of process parameters, etc.
The waveform is adjusted every 1/16 second if either the Add or Sub input is TRUE.
The amount added or subtracted is determined by the Step input.
For example, if Step has a value of 0.0625 and Add is TRUE,
the output will increase by 1.00 per sec
If neither Add nor Sub is TRUE, or if both are TRUE, the output holds its value.
Note that Step may be provided with a negative value, in which case Add causes the output to decreas
and Sub causes it to increase.
Preset overrides Step, Add and Sub, and forces the PVal value on to the output.
Clear overrides all other inputs and forces the output to 0.00
When the controller is initially switched on, the output of this function is set to 0.00
The Add, Sub, Preset or Clear inputs can be inverted.



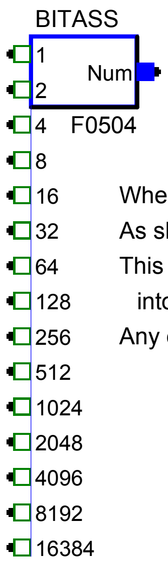
Type 20 - SAMPH - Sample and Hold

The Sample/Hold function takes a snapshot of a signal, and holds it indefinitely.
When the Samp input is TRUE, the output follows the input.
When Samp changes to FALSE, the output holds the last value sampled.
When the controller is initially switched on, the output of this function is set to 0.00
The Samp input can be inverted.



Type 1 - PREVSCAN

The PrevScan function outputs the value of the input signal 'x'
on the previous scan (i.e. 1/16 second ago.)
This enables the rate of change of a signal to be calculated.
If the difference between the input and output values is multiplied by 16,
we have the rate of change per second of the input signal.
This is useful for computing the D function in PD or PID control.
PREVSCAN blocks may be cascaded to form a shift register
to store signals over longer periods.

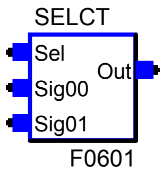


Type 46 - BITASS

BITASS is a bit assembler which takes 1's on the inputs and assembles
them into corresponding bits of the output Number.
This means that if the '1' input is TRUE, 1 is added to Num; if the '2' input
is TRUE, 2 is added to Num; and so on.
When first added to a diagram, the BITASS block has 2 inputs.
As shown here, the block can be expanded up to 15 inputs in all.
This BITASS function block is great for combining several digital signals
into a single composite signal.
Any of the inputs can be inverted.

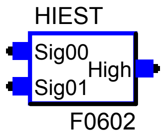
AmbiLogique Ltd

FUNCTION BLOCK LIST - 5 - ANALOGUE FUNCTIONS



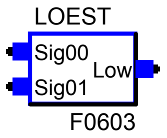
Type 21 - SELCT - Data Selector

The SELCT function routes one of it input signals through to the output.
The value on the Sel input is expected to be an integer.
If a fractional value is presented, it is rounded according to the rule:
if the fraction is 0.5 or more, take the next highest integer.
If Sel is presented with a value less than 0 or higher than the highest available signal,
the output is 0.00
Signal inputs can be added up to 14 in all.



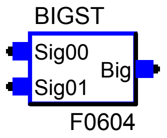
Type 22 - HIEST

The HIEST function picks the most positive signal on its inputs and routes it
to the output.
Inputs can be added up to 15 in all.



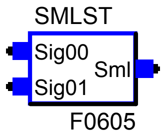
Type 23 - LOEST

The LOEST function picks the most negative signal on its inputs and routes it
to the output.
Inputs can be added up to 15 in all.



Type 24 - BIGST

The BIGST function picks the signal with the greatest magnitude (-2 beats +1)
and routes it to the output.
Inputs can be added up to 15 in all.



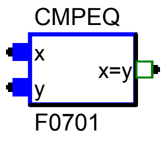
Type 25 - SMLST

The SMLST function picks the signal with the least magnitude (+1 is preferred to -2)
on its inputs and routes it to the output.
Inputs can be added up to 15 in all.

Signal selection is the primary mechanism for decision-based control in AmbiLogique PLC systems.

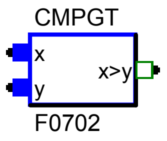
AmbiLogique Ltd

FUNCTION BLOCK LIST - 6 - SIGNAL SELECTION



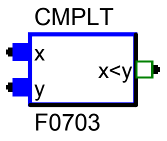
Type 26 - CMPEQ - Compare, output TRUE if equal

The CMPEQ function outputs TRUE if the two inputs are equal.
The output can be inverted to give TRUE if the inputs are unequal.



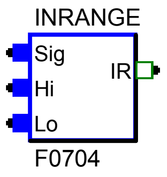
Type 27 - CMPGT - Compare, output TRUE if x greater than y

The CMPGT function outputs TRUE if the x input is more positive than y.
The output can be inverted to give TRUE if x is less than or equal to y.



Type 28 - CMPLT - Compare, output TRUE if x less than y

The CMPLT function outputs TRUE if the x input is less positive than y.
The output can be inverted to give TRUE if x is less than or equal to y.

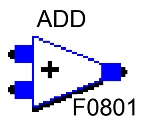


Type 45 - INRANGE - Ouput TRUE if signal within limits

The INRANGE function outputs TRUE if the Sig input is (less than or equal to Hi)
AND (greater than or equal to Lo).
Hi and Lo define a band within which the signal will generate a TRUE output.
Signals precisely equal to either boundary are considered to be acceptable,
and output TRUE.
The output can be inverted to give TRUE if Sig is outside the Lo-Hi range.

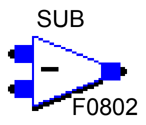
AmbiLogique Ltd

FUNCTION BLOCK LIST - 7 - COMPARISON



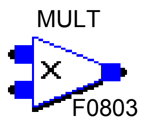
Type 29 - ADD

The ADD function output is the sum of all the input signals.
Inputs can be added up to 15 in all.



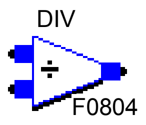
Type 30 - SUB (tract)

The SUB function output is the first input signal minus the second.
Inputs can be added up to 15 in all, in which case the signals on all the additional inputs are subtracted from the result.
Put another way, the output is the first input minus the sum of all the other inputs.



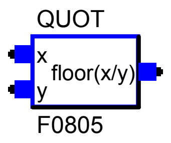
Type 31 - MULT (iply)

The MULT function output is the product of all the input signals.
Inputs can be added up to 15 in all.



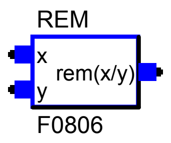
Type 32 - DIV (ide)

The DIV function output is the first input signal divided by the second.
Inputs can be added up to 15 in all, in which case
the signals on all the additional inputs are divided into the result.
Put another way, the output is the first input divided by the product of all the other inputs.



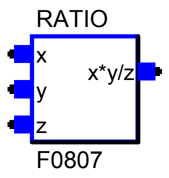
Type 33 - QUOT (ient)

The Quotient function outputs the largest integer which does not exceed x/y.
This is most useful in integer arithmetic, but the function works perfectly well with fractional numbers.



Type 34 - REM (ainder)

The Remainder function works out the quotient of x/y, i.e. the largest integer which does not exceed x/y.
It then multiplies the quotient by y, and subtracts the result from x.
The result is the remainder.
This is most useful in integer arithmetic, but the function works perfectly well with fractional num

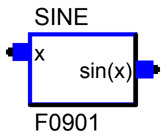


Type 35 - Ratio

The Ratio function multiplies x by y and divides the result by z.
Because it uses double precision for the multiply and divide, the result does not suffer from the rounding errors which could arise from the multiply and divide being done in two stages.
In addition, the Ratio function is much faster than the 2-stage process.

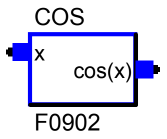
AmbiLogique Ltd

FUNCTION BLOCK LIST - 8 - BASIC ARITHMETIC



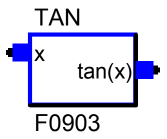
Type 36 - SINE

The Sine function outputs the sine of the input signal x.
Angles in AmbiLogique arithmetic are expressed in revolutions,
so that for example, 90° would be expressed as 0.25 revolution,
and the sine of 0.25 is therefore 1.0
If a number which is greater than 1 is supplied, the integer part is removed,
and the sine of the fraction is output.



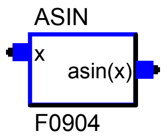
Type 37 - COS (ine)

The Cosine function outputs the cosine of the input signal x.
Angles in AmbiLogique arithmetic are expressed in revolutions,
so that for example, 90° would be expressed as 0.25 revolution,
and the cosine of 0.25 is therefore 0.0
If a number which is greater than 1.0 is supplied, the integer part is removed,
and the cosine of the fraction is output.



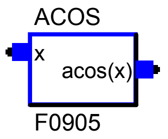
Type 38 - TAN (gent)

The Tangent function outputs the tangent of the input signal x.
Angles in AmbiLogique arithmetic are expressed in revolutions,
so that for example, 90° would be expressed as 0.25 revolution,
and the tangent of 0.25 is therefore infinity.
Note that infinity is a valid number (or set of numbers) in Ambilogique.
If a number which is greater than 1.0 is supplied, the integer part is removed,
and the tangent of the fraction is output.



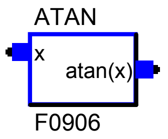
Type 39 - ASIN - arc sine

The ArcSine function outputs the arcsine of the input signal x.
Angles in AmbiLogique arithmetic are expressed in revolutions,
so that for example, the arcsine of 1.0 is 90°:
this would be expressed as 0.25 revolution.
If a number which is greater than 1.0 is supplied, the result is 0.25



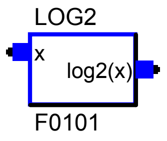
Type 40 - ACOS - arc cosine

The ArcCosine function outputs the arccosine of the input signal x.
Angles in AmbiLogique arithmetic are expressed in revolutions,
so that for example, the arccosine of 0.0 is 90°:
this would be expressed as 0.25 revolution.
If a number which is greater than 1.0 is supplied, the result is 0.00



Type 41 - ATAN - arc tangent

The ArcTan function outputs the arctangent of the input signal x.
Angles in AmbiLogique arithmetic are expressed in revolutions,
so that for example, the arctangent of 0.707 is 45°:
this would be expressed as 0.125 revolution.

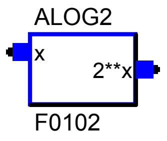


Type 42 - LOG2 - logarithm to base 2

The LOG2 function outputs the logarithm to base 2 of the input value.
If a value less than 0 is supplied, the negative sign is discarded.

Logarithms to other bases can be generated by multiplying:

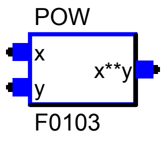
Base 10	0.30103
Base e	0.69315



Type 43 - ALOG2 - Antilogarithm to base 2 (2 to the power of x)

The ALOG2 function outputs the antilogarithm base 2 of the input value (2^value).
If an antilog to another base is required, the value should be pre-divided by:

Base 10	0.30103
Base e	0.69315



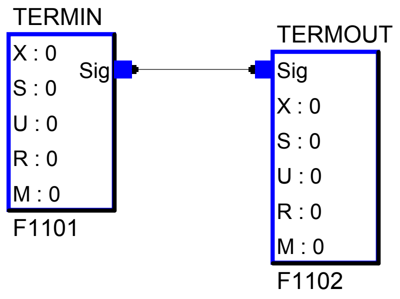
Type 44 - POW - x to the power of y

The POW function raises x to the power of y.
Either value can be fractional and of either sign.

Note that wires always run from the right-hand edge of an output to the left-hand edge of an input.
Any wire must be fed from one and only one output, but can connect to as many inputs as you wish.

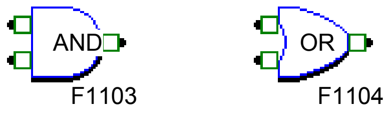
1. ANALOGUE PINS

ANALOGUE pins can carry any value, whether it be digital (TRUE/FALSE),
integer (...-2, -1, 0, +1, +2...), or real (including fractions and exponents).
Analogue pins are shown as solid blue squares.
Input and Output Terminals are examples of function blocks with analogue pins.



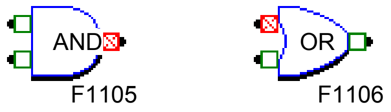
2. DIGITAL PINS

DIGITAL pins carry digital or Boolean signals which can have only the values TRUE and FALSE.
Digital pins are shown as hollow green squares.
Logic gates are examples of function blocks with digital pins.



DIGITAL pins can be inverted to change a TRUE signal to FALSE, and vice versa.

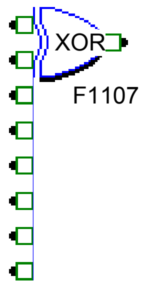
When this is done, the pin is shown as a red square with a diagonal cross.



Here, the AND gate has its output inverted and the upper input of the OR gate is inverted.
To invert a digital pin, right-click on the pin, then select "Invert pin"
from the pop-up menu.
To return an inverted pin to normal, right-click on the pin, then select "Invert pin"
from the pop-up menu.

3. ADDING AND DELETING PINS

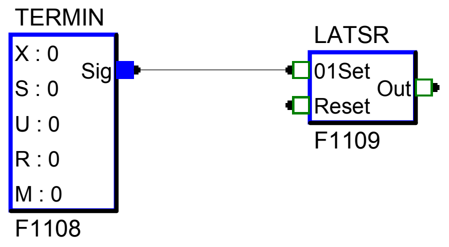
Some function blocks, including most logic gates, can be extended up to 14 inputs.
The additional pins are shown attached to a vertical bar which extends
below the function block.



This XOR gate has been extended to 8 inputs.
To add an input pin, right-click on any input pin, then select "Add pin"
from the pop-up menu.
To delete an input pin, right-click on any input pin, then select "Delete pin"
from the pop-up menu. The lowest pin will be deleted.
When you reach the maximum or minimum number of pins allowed for that
function block, the appropriate menu item is grayed out.

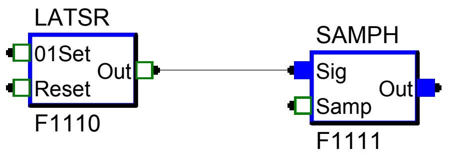
4. CONNECTING ANALOGUE AND DIGITAL PINS

AmbiLogique diagrams are TYPELESS, which means that all signals are compatible.
This means that an analogue pin can be wired to digital pins,
and a digital pin can be wired to analogue pins.



Here, an analogue pin from an input terminal
is wired to the digital input to the latch.

The rule is that if the signal is exactly zero, it is seen as FALSE by the digital input:
if the signal is non-zero, it is seen as TRUE.
If a broader definition of FALSE is required, to provide some tolerance in analogue systems,
please refer to the INRANGE function on diagram sheet 7.



Here, a digital output pin from a latch
is wired to the analogue input of a sample-hold

The rule is that if the digital signal is FALSE it is seen as 0.0000 by the analogue input:
if it is TRUE it is seen as 1.0000

AmbiLogique Ltd

FUNCTION BLOCK LIST - 11 - CONNECTION PINS