Function Sheet

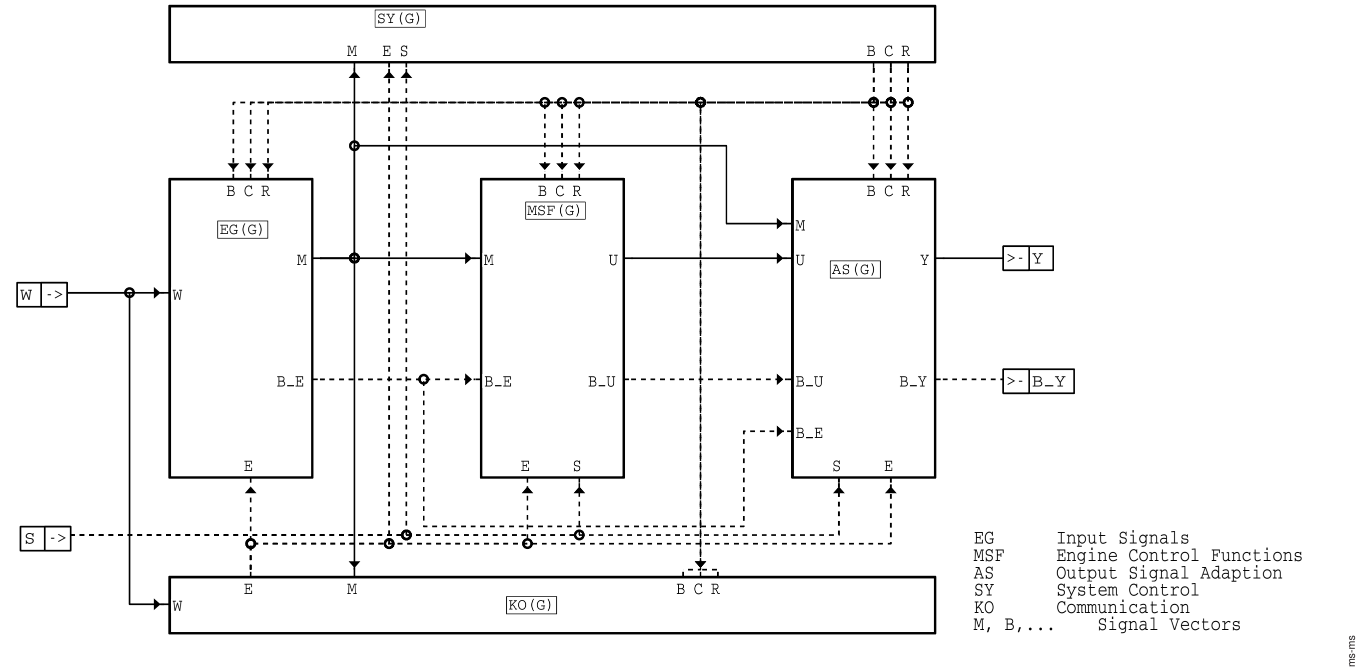
Extent shown:	
Blocks chosen:	ABK, APP, FB, FDEF, FW
System:	SG ME7.2
Project:	BMW 8-Zyl ME-7.2 M62LEV 3,5l und 4,4 l, variable Einl.-NW
Project code:	11/195;40
Responsible:	Markus Riemann
Department:	K3/EAF3
Phone:	20325
Date of issue:	19.04.2000

program release: 72111C_6200 (Predecessor : 72111C_6100)

Table of contents: Sections

MS 3.0 Engine control overview

FDEF MS 3.0 Function definition



ms-ms

ABK MS 3.0 Abbreviations

Variable	Source	Type	Description
B	MS	LOK	vector operating state conditions (only CU model)
B_E	MS	LOK	bit vector input variables
B_U	MS	LOK	bit vector manipulated variables
B_Y	MS	AUS	bit vector output variables
C	MS	LOK	vector controller state conditions (only CU model)
E	MS	LOK	Vector of error-flags
M	MS	LOK	vector engine variables physical (only CU model)
R	MS	LOK	vector computing base flags (only CU model)
S		EIN	vector switch flags (only CU model)
U	MS	LOK	revolution counter
W		EIN	vector converted input variables (only CU model)
Y	MS	AUS	vector of physical CU output variables

FB MS 3.0 Detailed description of function

The control unit model of the Motronic is made up of the following main parts:

- EG Input variables: Processing and diagnosis of the sensor signals, derivation of further variables
- SY System variables: Initializations, calculation cycles etc.
- MSF Engine control functions: Contain the functional groups to calculate the setting variables, e.g. torque coordination, injection, ignition, idle control etc.
- AS Output signals: Conversion of the setting variables into hardware-dependent setting signals
- KO Communication: Representation of the interfaces to the fault code memory management, customer service, driver, application etc.

The structure of the SG functions has been revised. The aim is a modular and more comprehensible structure. Checking and data flows are separated to a great extent. Data and checking flows are represented by different lines. Transfer of the data on the upper structure levels is performed by vectors.

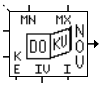
APP MS 3.0 Application hint

ASCETBLK 1.0 Description of ASCET block library

FDEF ASCETBLK 1.0 Function definition

Funktionsdarstellung:

Bei der Darstellung von Funktionen wird zwischen physikalischen Informationen (Datenfluß) und digitaler Steuerinformation (Kontrollfluß) unterschieden.
Datenfluß: Lastsignal, Drehzahl, Regelfaktor
Kontrollfluß: Bedingung Leerlauf, Schalter Fahrstufe, Fehler Kat
Durchgezogene Linien markieren den Datenfluß, gestrichelte Linien den Kontrollfluß.

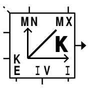


Grundblöcke (allgemeines):
- Bei Blöcken mit der Kennzeichnung "NOV" am Ausgang wird der Zustandswert des Blockes (Integratorinhalt, Flag, RAM-Zelle, etc.) im Dauer-RAM gespeichert (ansonsten im flüchtigen RAM). Im übrigen verhalten sich die Blöcke wie ihre Pendants ohne "NOV".
- Die Haupteingangs- und Hauptausgangswerte ("in" und "out") weisen im Block-Icon kein Symbol auf; sie sind mit 0.0 (float) bzw. FALSE (bool) vorbelegt, sofern nichts anderes angegeben ist.
- Nichtbeschaltete Eingänge sind mit 0.0 (float) bzw. FALSE (bool) vorbelegt, sofern nichts anderes angegeben ist.
- Bei einigen Blöcken kann an der linken oberen Ecke ein "Rastereingang" (default TRUE) angeschlossen werden, durch den die Berechnungshäufigkeit explizit festgelegt wird. Im folgenden bezeichnet "rasterZeit" den Abtand zwischen zwei Berechnungen.
- Eine Abweichung von der nachfolgenden Standardbelegung der Ein- und Ausgängen wird in der Beschreibung des Blockes angegeben.

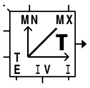
	Kürzel im Icon	Default-Wert	Bezeichnung
EINGÄNGE:	E	TRUE	Berechnung des Blocks freigeben
	I	FALSE	Initialisierung auslösen
	IV	0.0	Initialisierungswert
	K	0.0	hier: Integrationsfaktor K
	MX	1E35	obere Begrenzung der Ausgangsgröße
	MN	-1E35	untere Begrenzung der Ausgangsgröße

ascetblk-teil0

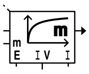
ascetblk-teil0



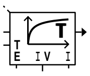
Integrator K
neuer Integratorwert := alter Integratorwert + K * rasterZeit * in
EINGÄNGE: K Integrationsfaktor



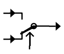
Integrator T
neuer Integratorwert := alter Integratorwert + (rasterZeit / T) * in
Der Minimalwert von T wird auf rasterZeit begrenzt.
EINGÄNGE: T Integrationszeitkonstante



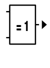
Rekursion
neuer Wert := alter Wert + m * (in - alter Wert)
EINGÄNGE: m Rekursionskonstante



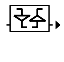
Tiefpass
neuer Tiefpasswert := alter Tiefpasswert + (rasterZeit / T) * (in - alter Tiefpasswert)
Der Minimalwert von T wird auf rasterZeit begrenzt.
EINGÄNGE: T Zeitkonstante



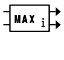
Eingangs-UmschalterUnten
Das Icon zeigt die Ruhestellung des Schalters, nichtbeschaltete Eingänge sind mit 0.0 vorbelegt.



Exklusiv-ODER
Der Ausgang wird TRUE, wenn genau ein Eingang TRUE ist.



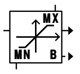
FlankeBi
Bei negativer oder positiver Flanke am Eingang, wird während dieses Simulationsschrittes am Ausgang TRUE ausgegeben. Sonst ist der Ausgang FALSE.



Maximum2
Am Ausgang liegt das Maximum der Eingangswerte an.
Der Ausgang i zeigt den Index des ersten Eingangs an, dessen Wert gleich dem ermittelten Maximum ist.

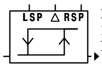
ascetblk-teil1

ascetblk-teil1



Begrenzer
Am Ausgang wird der auf den Bereich [MN, MX] begrenzte Eingangswert ausgegeben.
Ist eine Begrenzung aktiv, so wird der Ausgang B := TRUE gesetzt; ansonsten ist dieser Ausgang FALSE.

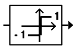
Betrag
Am Ausgang liegt der Betrag des Eingangswertes an.



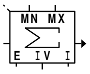
Hystrese
Der rechte und der linke Schaltpunkt der Hysterese ergibt sich aus der Beschaltung:

beschaltet	linker Schaltpkt	rechter Schaltpkt
LSP und delta	LSP	LSP + delta
LSP und RSP	LSP	RSP
delta und RSP	RSP - delta	RSP

Bei allen anderen Beschaltungen der Eingänge wird am Ausgang FALSE ausgegeben (fehlerhafte Beschaltung).



Signum
Ist der Eingangswert < 0.0, liegt am Ausgang der Wert -1.0, ansonsten der Wert 1.0 .



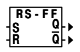
Akkumulator
Der Akkumulator wird um den Eingangswert additiv verändert und auf den Bereich [MN, MX] begrenzt.

FLAG
Nachbildung einer flüchtigen 1 Bit-Speicherzelle.

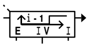
RAM
Nachbildung einer flüchtigen Speicherzelle.

ascetblk-teil2

ascetblk-teil2



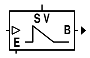
RS-FlipFlop
Das RS-FlipFlop hat einen Set-Eingang S und einen Reset-Eingang R.
Am Ausgang !Q liegt immer der zu Q invertierte Wert. Reset ist gegenüber Set dominant.



VerzögerungRaster
Verzögerung des Signals um ein Raster, d.h. out(i) := in(i-1).
Am Ausgang liegt der jeweils um einen Rastertakt verzögerte Wert an.
Wenn der Rastereingang offen ist, wird um einen Simulationstakt verzögert.

Ausschalt-Verzögerung
Der Ausgang folgt dem Schalten des Eingangs von TRUE nach FALSE nach der Verzögerungszeit, die am Eingang DELAY anliegt. Schaltet während der Verzögerung der Eingang wieder nach TRUE, liegt auch am Ausgang sofort TRUE an.

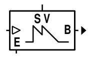
Einschalt-Verzögerung
Der Ausgang folgt dem Schalten des Eingangs von FALSE nach TRUE nach der Verzögerungszeit, die am Eingang DELAY anliegt. Schaltet während der Verzögerung der Eingang wieder nach FALSE, liegt auch am Ausgang sofort FALSE an.



Timer
Eine positive Flanke am Eingang bewirkt, daß der Timer gestartet wird, d.h.
- der interne Timer wird auf den Wert (in Sekunden) gesetzt, der am Eingang SV anliegt,
- der Ausgang wird TRUE und bleibt TRUE bis der Timer abgelaufen ist.

Eine erneute positive Taktflanke am Eingang hat keine Auswirkung, solange der Timer noch nicht abgelaufen ist. Liegt an E FALSE, wird der Timer gestoppt, bis E wieder TRUE ist.

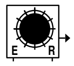
EINGÄNGE:	in	Starten des Timers
	SV	Timerzeit
AUSGÄNGE:	B	Timer läuft



Timer-Retrigger
Grundfunktion wie "Timer", jedoch: Eine erneute positive Taktflanke am Eingang bewirkt stets Neustart des Timers.

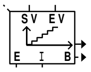
ascetblk-teil3

ascetblk-teil3



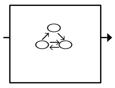
ZeitZähler
TRUE am Eingang R setzt den Zeitzähler auf 0.0 zurück. Wird R = FALSE, beginnt der ZeitZähler zu laufen.
Liegt an E FALSE, so wird der Zeitzähler gestoppt. Der Zeitzähler zeigt die abgelaufene Zeit in Sekunden an.

EINGÄNGE:	R	Rücksetzen des ZeitZählers



Zähler
Dieser Block zählt in jedem Simulationsschritt um eins aufwärts bzw. abwärts. Startwert, Endwert und damit die Zählrichtung werden festgelegt, wenn am Eingang I TRUE anliegt.
Wenn der Wert von SV größer als der Wert von EV ist, dann wird abwärts (ansonsten aufwärts) gezählt, bis der Endwert erreicht ist. Das Erreichen des Endwertes wird durch ein TRUE am Ausgang B angezeigt.
Der Zähler kann mit dem Eingang E gestoppt werden.

EINGÄNGE:	SV	Startwert des Zählers
	EV	Endwert des Zählers
	I	Zähler starten
AUSGÄNGE:	B	Endwert erreicht



Zustandsautomat
Der Kontrollfluß wird durch logische Gatter und Zustandsautomaten dargestellt. In Zustandsautomaten wird der Funktionsablauf in graphischer Form mit Hilfe von "Zuständen" und "Übergängen" abgebildet.

Zustand: (Ellipse)	Innerhalb eines Zustandsautomaten ist jeweils genau ein Zustand aktiv, d.h. die zu diesem Zustand gehörenden Aktionen werden ausgeführt. Der Name des Zustandes ist innerhalb der Ellipse dargestellt.
Übergang: (Pfeil)	Der Übergang von einem Zustand zum anderen erfolgt, wenn die Übergangsbedingung erfüllt ist. Dabei werden diesem Übergang zugeordnete Aktionen ausgeführt. Die Bedingung, die erfült sein muß, damit ein Übergang stattfindet, steht neben dem jeweiligen Pfeil; ggf. steht nur ein logischer Name für die Bedingung und die ausfürliche Beschreibung ist dem nachfolgenden Text zu entnehmen. Bevorzugt wird die Bedingung mit der niedrigsten Nummer.

Für jeden Zustandsautomaten ist festgelegt, welcher Zustand beim Start des Automaten angenommen werden soll (S) und welcher Zustand bei erfüllter RESET-Bedingung (R).

ascetblk-teil4

ascetblk-teil4

ABK ASCETBLK 1.0 Abbreviations

Variable	Source	Type	Description
N		EIN	Engine speed

FB ASCETBLK 1.0 Detailed description of function

APP ASCETBLK 1.0 Application hint

ASCETSDB 1.23 ASCET-SD descripton of block library

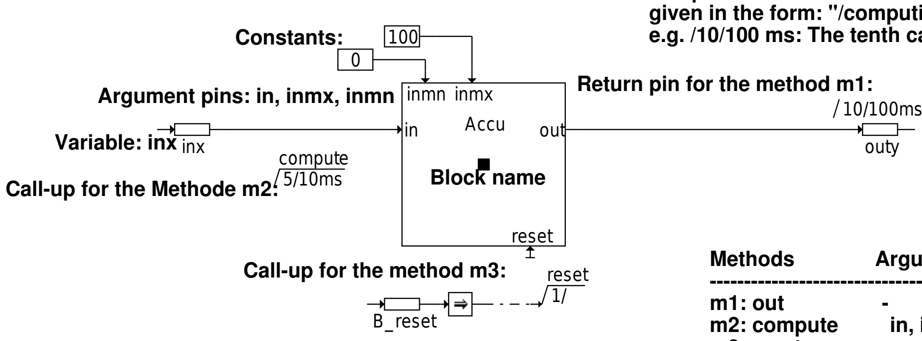
FDEF ASCETSDB 1.23 Function definition

Graphical presentation of basic elements

Basic elements are presented in the diagram as rectangular blocks. Communication between basic elements is displayed by connecting lines. The interfaces between basic elements are the pins at the block edges. Each block has return pin that outputs the result from the block. In addition to this, there are argument pins that provide the inputs into the block as well as method pins that are used for those methods without input arguments and without return values.

The methods call up functions in the block.

The process information and the computing sequence are given in the form: "/computing sequence/process", e.g. /10/100 ms: The tenth call-up in the 10-ms computing frame



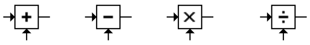
Methods	Arguments	Return value
m1: out	-	Float
m2: compute	in, inmx, inmn	-
m3: reset	-	-

The example given above shows a block with 3 methods:
- The method m1 "out" has one return value
The method "out" is called up by the request for the return value from the subsequent block outy, that is in the tenth position in the computing sequence in the 100-ms computing frame.
- The method m2 "compute" has three arguments (in, inmn, inmx) yet no return value.
The method "compute" is called up at the fifth position in the computing sequence in the 10-ms computing frame.
- The method m3 "reset" has neither arguments nor a return value. This is therefore represented by the "method pin".
If B_reset is true, then the method "reset" is called up first (1/) in the computing sequence.

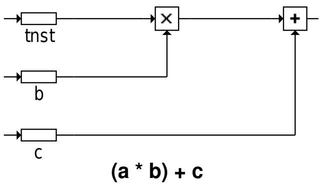
ascetsdb-e-beschrei

ascetsdb-e-beschrei

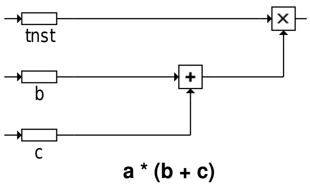
Arithmetic operations



Equations can be described using arithmetic operations (addition, subtraction, multiplication and division). Equations are represented graphically such that the return value of an operation is the argument for the subsequent operation.

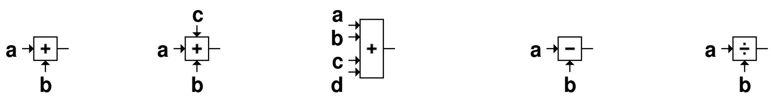


(a * b) + c

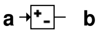


a * (b + c)

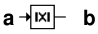
The arguments of primitive operations and their computing sequence are shown in the following:



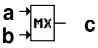
a + b | **a + b + c** | **a+b+c+d** | **a - b** | **a / b**



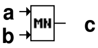
Negation: b = - a



Amount: b = |a|



Maximum for input values: c = MAX(a,b)

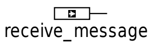


Minimum for input values: c = MIN(a,b)

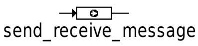
ascetsdb-e1-artihme

ascetsdb-e1-artihme

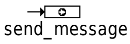
Variables



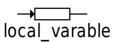
Receive message are input variables of the function that are made available from another function.



Send/Receive variables are output variables of the function that are used both within as well as outside of the function.



Send messages are output variables of the function and are available for the other functions.



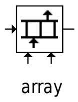
Local variables are only made available and used within the function.

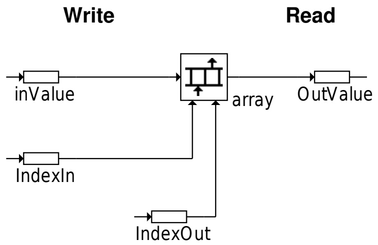
ascetsdb-e2-variabl

ascetsdb-e2-variabl

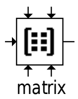
Arrays and matrices

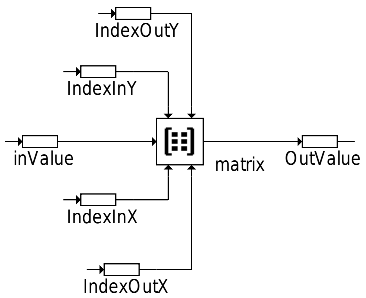
Arrays and matrices have two methods for reading and writing access to the elements. The reading and writing can be independent of one another.





Array:
- The value to be written is connected to the left pin, the associated index to the lower left pin.
- The value to be read is connected to the right pin, the associated index to the lower right pin.





Matrix:
Matrices behave as arrays, whereby the methods here have two index arguments (x, y):
- The index x is connected to the lower left, the index y to the upper left pin for writing access.
- The index x is connected to the lower right, the index y to the upper right pin for reading access.

ascetsdb-e3-arrays-

ascetsdb-e3-arrays-

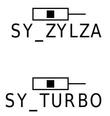
Constants

Boolean Constants

ascetsdb-e4-konstan

ascetsdb-e4-konstan

System constants



System constants are constants that are permanently anchored in the program and cannot be applied. System constants can switch functional positions conditionally on or off.

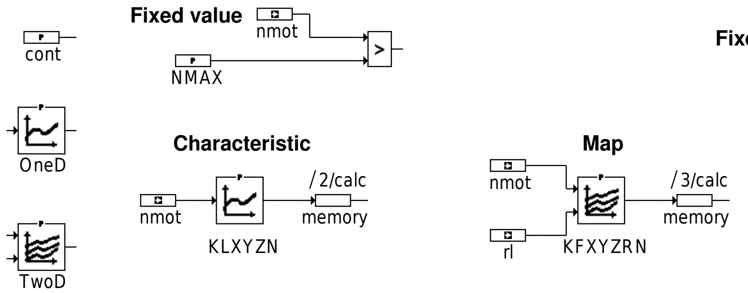
Example

SY_ZYLZA: Cylinder number
SY_TURBO: Engine with or without turbo-charger

ascetsdb-e5-systemk

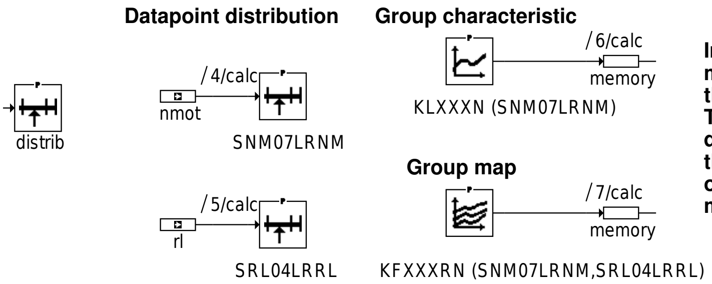
ascetsdb-e5-systemk

Fixed values, Characteristics, Maps, Group characteristics, Group maps and datapoint distribution



Fixed values are applicable parameters

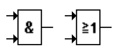
Characteristics have one argument.
Maps have two arguments as the input.
Both have one return value

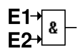


In the case of group characteristics and group maps, several characteristics and maps pick up the same datapoint distributions.
To this end, the current datapoint from the datapoint distribution, e.g. SNM07LRNM, is first computed from the dependent parameter e.g. nmot. The computation of the output value for the group characteristic or map then follows using this current datapoint.

ascetsdb-e6-kl-kf-g

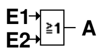
Bit operations





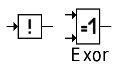
AND element:

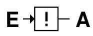
E1	E2	A
0	0	0
1	0	0
0	1	0
1	1	1



OR element:

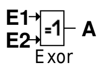
E1	E2	A
0	0	0
1	0	1
0	1	1
1	1	1





Negation

E	A
0	1
1	0



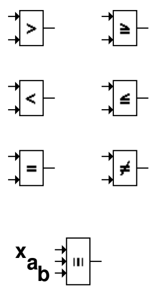
Exklusive OR:

E1	E2	A
0	0	0
1	0	1
0	1	1
1	1	0

ascetsdb-e7-bitoper

Comparator

The comparator provides TRUE at the output if the comparison applies.
If the comparison is not fulfilled, then FALSE is given as the output.

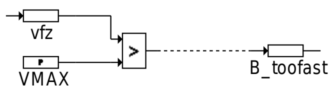


Greater, greater than or equal to

The comparison is always read from top to bottom (interval excepted):

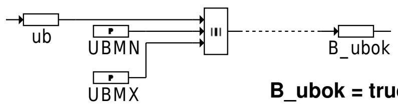
Less, less than or equal to

If vfz is greater than VMAX, then the condition B_toofast is TRUE



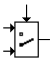
Equal, unequal

Closed interval:
a <= x <= b



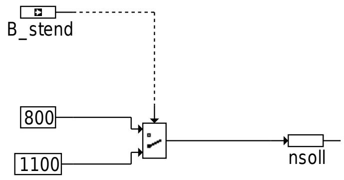
B_ubok = true if UBMN <= ub <= UBMX

ascetsdb-e8-verglei



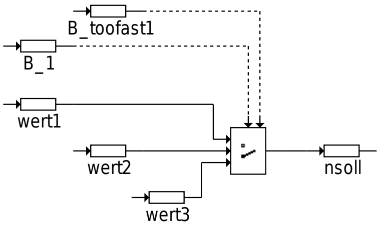
Multiplex operator "Multiplexer", "Switch"

A multiplexer switches a value through to the output as a dependency of the input conditions.
The multiplexer icon is shown in the dormant position, i.e. if the input conditions are false.



Example "Simple multiplexer"
- if B_stend = false: nsoll = 1100
- if B_stend = true: nsoll = 800

An input condition is assigned to each value in the case of cascaded multiplexers. The uppermost value, the input condition for which is true, is switched through. If there is no input condition of true, then the lowest value is switched through.

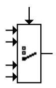


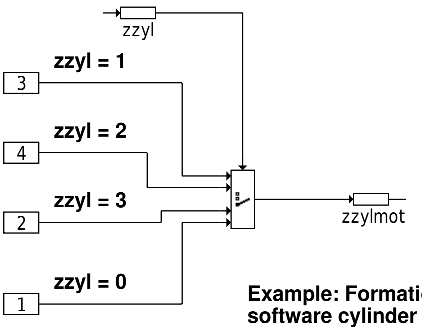
Example "Mehrfach-Muxer":
- if B_1 = true: nsoll = value1
- if B_1 = false & B_2 = true: nsoll = value2
- if B_1 = false & B_2 = false: nsoll = value3

ascetsdb-e9-multipl

ascetsdb-e9-multipl

CASE operator





The CASE operator switches one of the remaining left inputs through to the output as a dependency of a top-applied discrete control value (1, 2, 3, ...).
If the control value 1 is the first value, then 2 is the second value and is switched through immediately.
If the value is outside of the range, then the lowest input (default) is switched through.

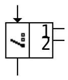
Example: Formation of the physical cylinder number onto the software cylinder number.

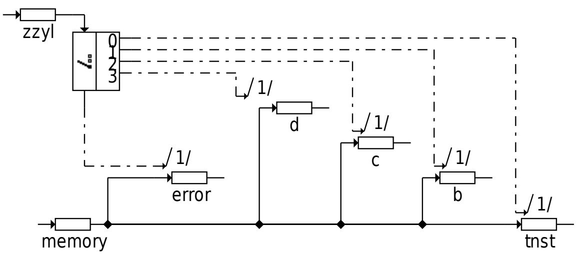
ascetsdb-e10-case-o

ascetsdb-e10-case-o

Switch

The SWITCH operator activates the matching control flows over the right-hand outputs as a dependency of an upper applied discrete control value (1, 2, 3, ...). If there is no matching output existing, then the control flow is activated at the lower output.



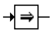


Example:
One of the following operations is executed depending on zzyl:
- if zzyl = 0: a = memory
- if zzyl = 1: b = memory
- if zzyl = 2: c = memory
- if zzyl = 3: d = memory
- otherwise: error = memory

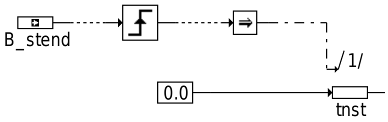
ascetsdb-e11-kontro

ascetsdb-e11-kontro

If .... then

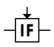


The If .. Then operation analyzes a logic condition and activates all computing sequences for TRUE that are connected to the control flow. The computing sequence is defined by the numbering.

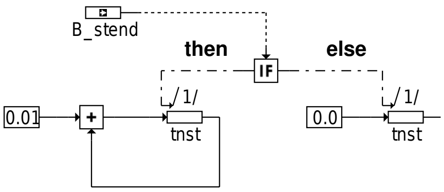


Example:
If B_stend changes to true, then tnst = 0 is set.

If

then  **else**

The IF .. Then .. Else operation analyzes a logic condition and activates all computing sequences of the then control branch for TRUE, and all computing sequences of the else control branch for FALSE. The computing sequence at each control branch is defined by the numbering.



Example: If B_stend = true,
then tnst is incremented by 0.01 sec.
in the 10-ms time frame.
Otherwise (B_stend = false) tnst = 0 is set.

Mule bridge for IF:

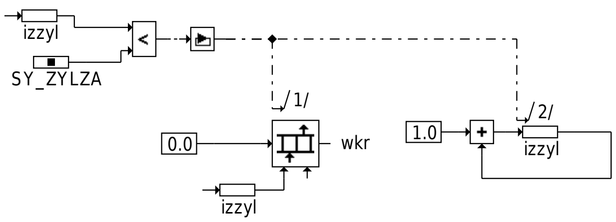
I=True <-- IF --> F=False

ascetsdb-e12-kontro

ascetsdb-e12-kontro

While loop

The computing sequence within the control flow of the loop is executed for such a time as the input condition is fulfilled, i.e. is TRUE. The loop is aborted when the input condition is FALSE. The value for terminating the While loop is normally formed within the loop. This usually concerns a counter here that shall count up to a certain value.

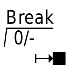


Example:
The array wkr[i] is written with 0 for such a time as izzy < SY_ZYLZA. Each element of the array is initialized with 0 by the numerical variable izzyl at the index input of the array.

ascetsdb-e13-while

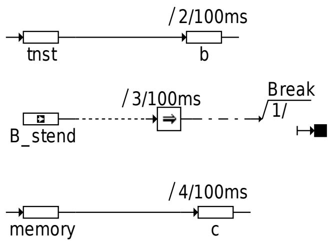
ascetsdb-e13-while

Break



The break operation prematurely interrupts a process, e.g. the functions component in a computing frame. All subsequent calculations of the function in the process with a higher number for the computing sequence are not then executed.

Example:



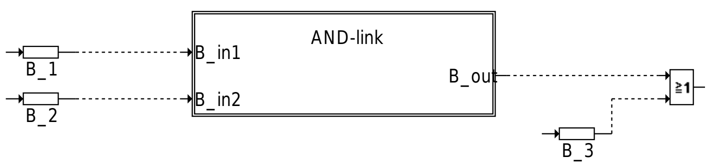
In accordance with the computing sequence, a break is triggered after the operation b = a exactly then when B_stend = TRUE. The 100-ms process is started if a break occurs. The subsequent operation, c = memory, is no longer executed.

ascetsdb-e14-kontro

ascetsdb-e14-kontro

Hierarchy:

The hierarchy is graphical possibility for structuring functions.
The hierarchy block is identified by a double border.
The corresponding hierarchy level is identified by the name, here "AND link".
The transfer element is only a designator for the links between both levels.

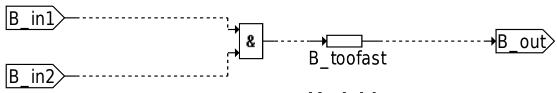


ascetsdb-e15-hierar

ascetsdb-e15-hierar

Function in the graphical "AND link" hierarchy:

Designator for input links



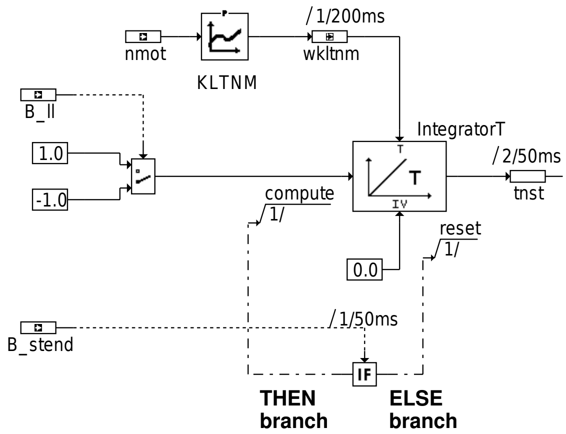
Variable

Designator for return links

ascetsdb-and-link

ascetsdb-and-link

Example



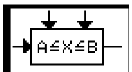
The method "reset" of the integrator is executed in the 50-ms computing frame as long as the condition B_stend = FALSE. This method causes the internal memory cell of the integrator to initialized with the IV-value, i.e. 0.0. If the condition now becomes B_stend = TRUE, then the left-hand control flow is activated and the method "compute" has as the arguments time T and the input value. This depends on B_ll. The input value = 1.0 for B_ll = TRUE; the input value = -1.0 for B_ll = FALSE. The number in front of the computing frame indicates the computing sequence: The time constant T is computed in the 200-ms computing frame and the message wkltnm is stored in Send/Receive. The IF .. THEN .. ELSE query is first executed in the 50-ms computing frame. The integration value is written into the variable a in the second step.

ascetsdb-e-b-beispi

ascetsdb-e-b-beispi

ASCET-SD System Library

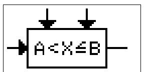
Comparators



ClosedInterval

`ClosedInterval` returns TRUE, if the value `x` is in the closed interval defined by A and B.

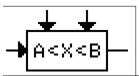
Methods	Characteristics	Arguments	Return Value
out	TRUE is returned, if `A <= x <= B`. Otherwise FALSE is returned.	x::continuous A::continuous B::continuous	TRUE or FALSE



LeftOpenInterval

`LeftOpenInterval` returns TRUE, if the value `x` is in the left open interval defined by A and B.

Methods	Characteristics	Arguments	Return Value
out	TRUE is returned `A < x <= B. Otherwise FALSE is returned`.	x::continuous A::continuous B::continuous	TRUE or FALSE



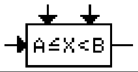
OpenInterval

`OpenInterval` returnes TRUE, if the value `x` is in the open interval defined by A and B .

Methods	Characteristics	Arguments	Return Value
out	TRUEis returned, if `A < x < B`. Otherwise FALSE is returned .	x::continuous A::continuous B::continuous	TRUE or FALSE

ascetsdb-etassyse1

ascetsdb-etassyse1



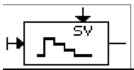
RightOpenInterval
`RightOpenInterval` returnes TRUE, if the value `x` is in the right open interval defined by A and B.

Methods	Characteristics	Arguments	Return Value
out	TRUE is returned, if `A <= x < B`.Otherwise FALSE is returned.	x::continuous A::continuous B::continuous	TRUE or FALSE

GreaterZero
`GreaterZero` returns TRUE, if the value x is greater than zero.

Methods	Characteristics	Arguments	Return Value
out	TRUE is returned, if `x > 0.0`. otherwise FALSE is returned.	x:continuous	TRUE or FALSE

Counter & Timer

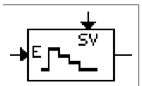


CountDown
CountDown decrements the counter and signals when the counter has reached zero.

Methods	Characteristics	Arguments	Return Value
start	The counter is set to the start value .	startValue::unsigned discrete	none
compute	The counter is decremented by one.	none	none
out	TRUE is returned if the counter is greater than zero. Otherwise FALSE is returned.	none	TRUE or FALSE

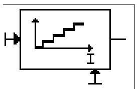
ascetsdb-etassyse2

ascetsdb-etassyse2



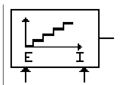
CountDownEnabled
CountDownEnabled dercrements the counter and signals when the counter has reached zero. This counter must be enabled explicitly.

Methods	Characteristics	Arguments	Return Value
start	The counter is set to start value.	startValue:: unsigned discrete	none
compute	If enable isTRUE, the counter is decremented by one.	enable::TRUE or FALSE	none
out	TRUE is returned if the counter is greater zero.Otherwise FALSE is returned.	none	TRUE or FALSE



Counter
Counter increments the couner by one.

Methods	Characteristics	Arguments	Return Value
reset	The counter is set to zero.	none	none
compute	The counter is incremented by one .	none	none
out	The counter value is returned .	none	unsigned discrete

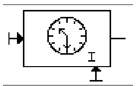


CounterEnabled
CounterEnabled increments the counter by one. This counter must be enabled explicitly.

Methods	Characteristics	Arguments	Return Value
reset	If `initEnable is` TRUE, the counter is set to zero .	initEnable::TRUE or FALSE	none
compute	If `enable` is TRUE, the counter is incremented by one.	enable::TRUE orFALSE	none
out	The counter value is returned.	none	unsigned discrete

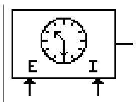
ascetsdb-etassyse3

ascetsdb-etassyse3



StopWatch
StopWatch increments the time counter by one dT.

Methods	Characteristics	Arguments	Return Value
reset	The counter is set to zero.	none	none
compute	The time counter is inremented by dT.	none	none
out	The time counter value, i. e. the time elapsed since the last start, is returned .	none	continuous

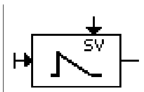


StopWatchEnabled
StopWatchEnabled increments the time counter by one dT. Thies timer must be enabled explicitly.

Methods	Characteristics	Arguments	Return Value
reset	If `initEnable is` TRUE, the internal counter is set to zero.	initEnable::TRUE or FALSE	none
compute	If `enable` is TRUE, the time counter is incremented by dT .	enable::TRUE or FALSE	none
out	The time counter value, i.e. the time elapsed since the last start and while `enable` was TRUE is returned.	none	continuous

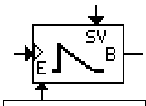
ascetsdb-etassyse4

ascetsdb-etassyse4



Timer
Timer decrements the time counter by dT and singals when the time counter has reached zero . Its not retriggerable.

Methods	Characteristics	Arguments	Return Value
start	The time counter is set to zero, if the time counter value was previously less than or equal to zero.	startTime::continuous	none
compute	The time counter is decremented by dT.	none	none
out	TRUE is returned, if the time counter is greater then zero. Otherwise FALSE is returned.	none	continuous

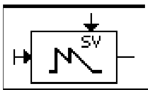


TimerEnabled
TimerEnabled decrements the time counter by dT and signals when the time counter reaches zero. It must be enabled explicitly.

Methods	Characteristics	Arguments	Return Value
compute	If `enable` is TRUE, `in` has a rising edge and the time counter value is less or equal zero, the timer is started, i.e. its counter value is set to the start time. Otherwise the time counter is decremented by dT. If `enable` is FALSE nothing happens.	enable::TRUE or FALSE in::TRUE or FALSE startTime::continuous	none
out	TRUE is returned, if the time counter value is greater than zero. Otherwise , FALSE is returned .	none	continuous

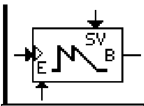
ascetsdb-etassyse5

ascetsdb-etassyse5



TimerRetrigger
TimerRetrigger decrements the time counter by dT and signal when the time counter has reached zero.It can be retriggered.

Methods	Characteristics	Arguments	Return Value
start	The time counter is set to the start value	startTime::continuous	none
compute	The time counter is decremented by dT.	none	none
out	TRUE is returned, if the time counter value is greater than zero.Otherwise, FALSE is returned.	none	continuous



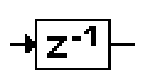
TimerRetriggerEnabled
TimerRetriggerEnabled decrements the time counter by dT and signals when time counter has reached zero. It can be retriggered and must be enabled explicitly.

Methods	Characteristics	Arguments	Return Value
compute	If `enable` is TRUE and it has a rising edge,the timer is started i.e. its counter value is set to the start value. Otherwise the time counter is decremented by dT (the time frame). If enable is FALSE nothing happens.	enable::TRUE or FALSE in::TRUE or FALSE startValue::continuous	none
out	TRUE is returned, if the time counter value is greater than zero. Otherwise FALSE is returned .	none	continuous

ascetsdb-etassyse6

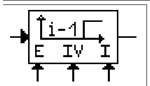
ascetsdb-etassyse6

Delay



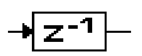
DelaySignal
DelaySignal delays its input signal by one evaluation step.

Methods	Characteristics	Arguments	Return Value
compute	The input signal is buffered	signal::TRUE or FALSE	none
out	The buffered signal is returned ,thus the input signal is delayed by one step.	none	TRUE or FALSE



DelaySignalEnabled
DelaySignalEnabled delays its input singal by one evaluation step. It must be enabled explicitly.

Methods	Characteristics	Arguments	Return Value
reset	If `initEnable` is TRUE, `initValue` is buffered	initEnable::TRUE or FALSE initValue::TRUE or FALSE	none
compute	If `enable` is TRUE, the input signal is buffered.	signal::TRUE or FALSE enable::TRUE or FALSE	none
out	The buffered signal is returned, thus the input signal is delayed by one step..	none	TRUE or FALSE

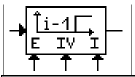


DelayValue
DelayValue delays its input value by one evaluation step.

Methods	Characteristics	Arguments	Return Value
compute	The input value is buffered .	value:continuous	none
out	The buffered value is returned, thus the input value is delayed by one step.	none	continuous

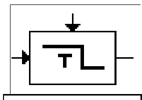
ascetsdb-etassyse7

ascetsdb-etassyse7



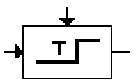
DelayValueEnabled
DelaySignalEnabled delays its input value by one evaluation step. It must be enabled explicitly.

Methods	Characteristics	Arguments	Return Value
reset	If `initEnable` is TRUE,`initValue` is buffered.	initEnable::TRUE or FALSE initValue::continuous	none
compute	If `enable` is TRUE, the input value is buffered .	value::continuous enable::TRUE or FALSE	none
out	The buffered value is returned, thus the input value is delayed by one step	none	TRUE or FALSE



TurnOffDelay
TurnOffDelay delays a falling edge of the input signal .

Methods	Characteristics	Arguments	Return Value
compute	A fallling edge of the input signal is delayed. If the signal flips from TRUE to FALSE, a timer is started . On being FALSE the timer is incremented by dT and is compared to delayTime. If the input signal is TRUE, the timer is reset.	signal::TRUE or FALSE delayTime::continuous	none
out	TRUE is returned if the input signal is TRUE or the timer has not exceeded delayTime. Otherwise FALSE is returned.	none	TRUE or FALSE



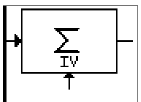
TurnOnDelay
TurnOffDelay delays a rising edge of the input signal.

Methods	Characteristics	Arguments	Return Value
compute	A rising edge of the input signal is delayed. If the signal flips from FALSE to TRUE a timer is started. On being TRUE the timer is incremented by dT and is compared to delayTime. If the input signal is FALSE, the timer is reset.	signal::TRUE or FALSE delayTime::continuous	none
out	FALSE is returned if the input signal is FALSE, or the timer has not exceeded delayTime . Otherwise,TRUE is returned.	none	TRUE or FALSE

ascetsdb-etassyse8

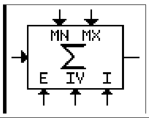
ascetsdb-etassyse8

Memory



Accumulator
Accumulator adds up its input value.

Methods	Characteristics	Arguments	Return Value
reset	The accumulator is set to initValue.	initValue::continuous	none
compute	The accumulator is incremented by the input value, i.e. `accumulator (new) = accumulator (old) + input value`.	value::continuous	none
out	The accumulator value is returned.	none	continuous



AccumulatorEnabled
AccumulatorEnabled adds up its input value. It must be enabled explicitly and its accumulator value can be limited.

Methods	Characteristics	Arguments	Return Value
reset	If `initEnable` is TRUE, the accumulator value is set to `initValue`.	initValue:continuous initEnable::TRUE or FALSE	none
compute	If `enable` is TRUE, the accumulator is incremented by the input value, i.e. `accumulator(new) = accumulator(old) + input value`. Additionally, the accumulator value is limited by `mn` and `mx`.	value:continuous mn:continuous mx::continuous enable::TRUE or FALSE	none
out	The accumulator value is returned.	none	continuous

ascetsdb-etassyse9

ascetsdb-etassyse9