Isc - Idle Speed Control

1 [Idle Speed Control] Idle Speed Control

1.1 [Overview]

Figure 1: [Isc Function Overview]

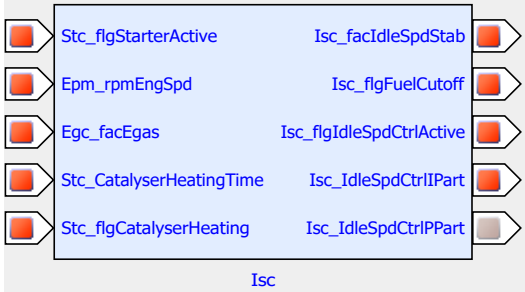
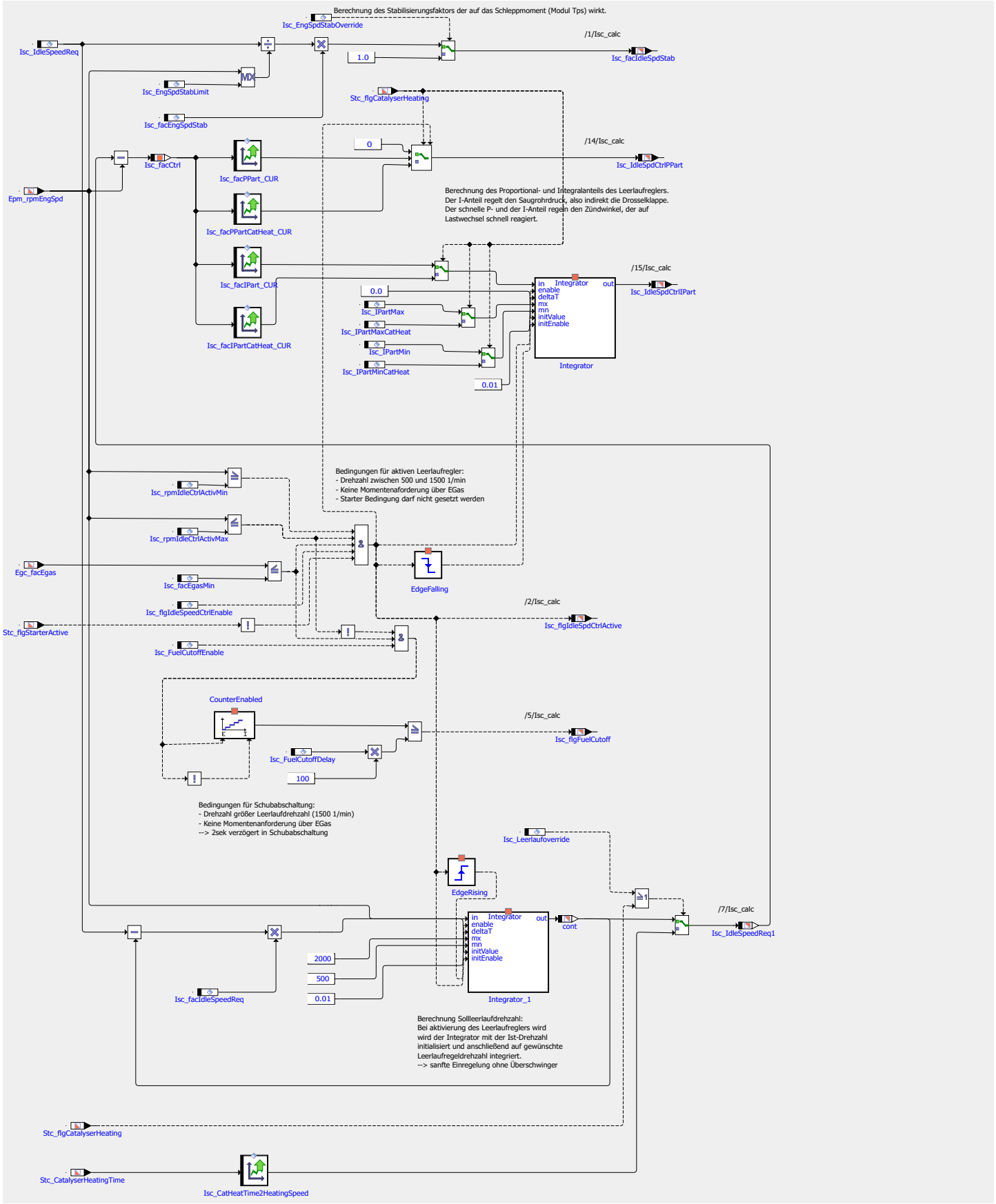


Figure 2: [IdleSpeedController.Main]



1.2 Module der Motorsteuerung [Module der Motorsteuerung]

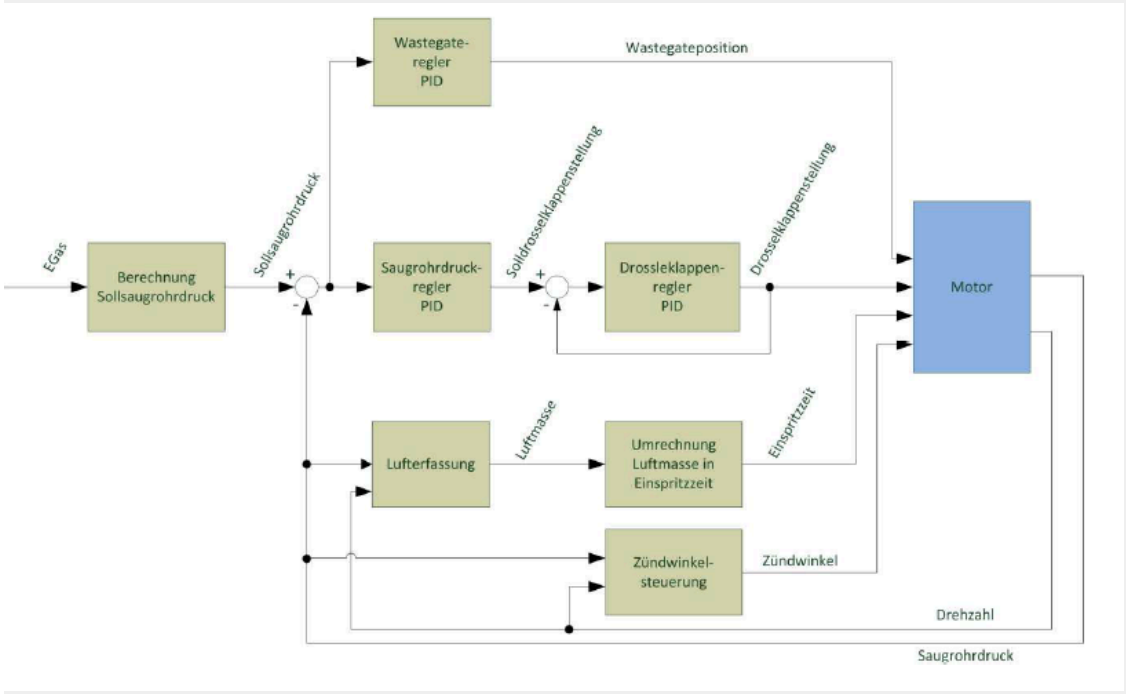
In der Motorsteuerung sind die grundlegenden Module zum Betreiben eines Motors vorhanden. In diesem Kapitel werden neu implementierte Module beschrieben sowie Änderungen an bereits Vorhandenen. Zum Beispiel werden

die zusätzlich benötigten Sensoren und Aktuatoren sowie weitere Module in die Motorsteuerung integriert. Das vorhandene Modell wird in Kapitel 2.8 genauer beschrieben.

1.2.1 Regler Übersicht [Regler Übersicht]

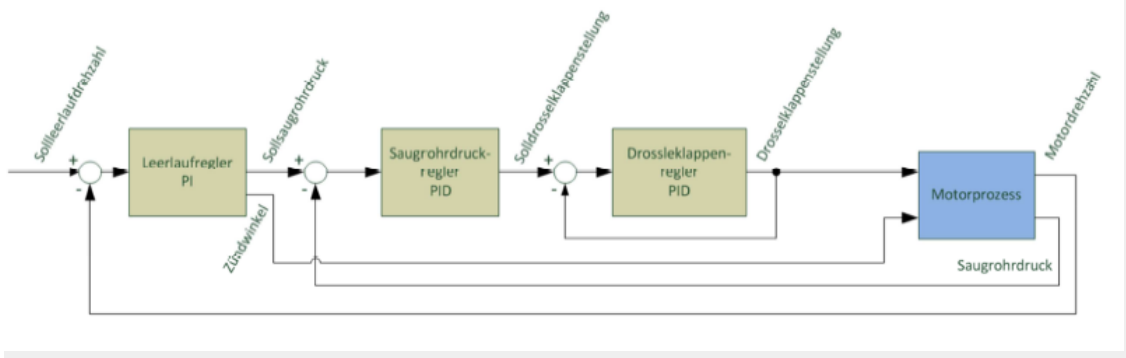
Die Motorsteuerung hat die Aufgabe bestimmte Betriebszustände zu regeln. Dafür sind mehrere Regler notwendig, die in reihe und parallel geschaltet sind.

Figure 3: Reglerstruktur im Normalbetrieb [isc_png_1]



In Abbildung 4-1 ist eine Übersicht über die Reglerstruktur des Luftpfades dargestellt, sowie die Berechnung der Einspritzzeit und des Zündwinkels. Damit wird im Normalbetrieb aus dem Momentenwunsch des Fahrers ein Sollsaugrohrdruck berechnet. Auf Grund dieses Saugrohrdruckes berechnet die Motorsteuerung mit Hilfe des Saugrohrdruckreglers eine Solldrosselklappenstellung, welche dann vom Drosselklappenregler eingeregelt wird. Wird ein größerer Saugrohrdruck benötigt als saugmotorisch bereitgestellt werden kann, wird der Wastegateregler aktiv und die Drosselklappe wird 100% geöffnet. Über die Stellung des Wastegates (Bypassventil des Turboladers) wird der Saugrohrdruck geregelt.

Figure 4: Reglerstruktur bei aktivem Leerlaufregler [isc_png_2]



Ist der Motor in Leerlaufbetrieb (Abbildung 4-2), wird die Sollleerlaufdrehzahl mit Hilfe des Leerlaufreglers geregelt. Der Regler hat dabei zwei Stellgrößen die beeinflusst werden. Zum einen wird der Saugrohrdruck über die Drosselklappenstellung beeinflusst und zum anderen wird der Zündwinkel und hiermit das Moment beeinflusst.

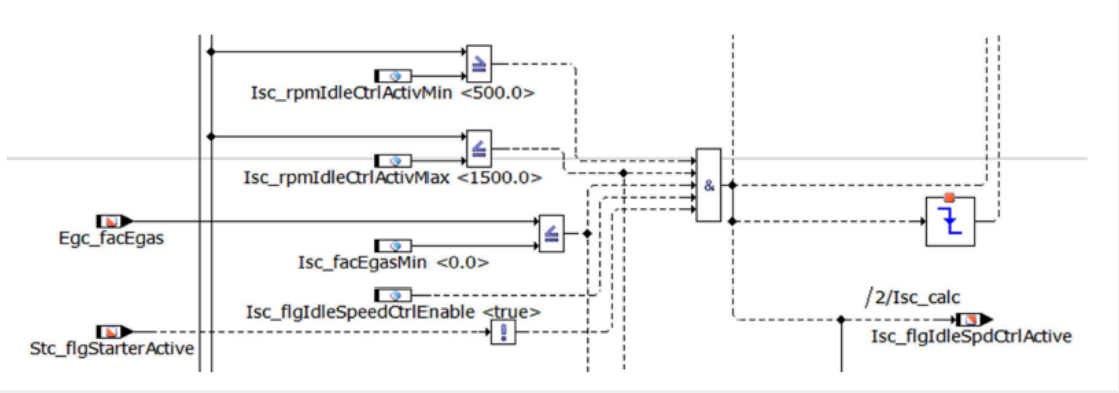
1.2.2 Leerlaufregler [Leerlaufregler]

Der Leerlaufregler befindet sich im Funktionsmodul IdleSpeedController (Isc). In diesem Modul werden die Korrekturgrößen berechnet, welche zur Regelung der Leerlaufdrehzahl notwendig sind. Außerdem werden die Bedingungen für den aktiven Leerlaufregler abgefragt.

Bedingungen für aktive Leerlaufregelung:

Die Leerlaufregelung darf nur in bestimmten Betriebszuständen des Motors aktiv sein. In der Motorsteuerung sind daher Bedingungen zu erfüllen. Zum einen ist die Regelung nur aktiv, sobald die Drehzahl in einem bestimmten Bereich liegt. Mit den Parametern Isc_rpmIdleCtrlActivMin und Isc_rpmIdleCtrlActivMax sind diese Grenzen parametrierbar. Zum anderen wird mit dem Parameter Isc_facEgasMin die maximale Gaspedalstellung, bei der die Regelung noch aktiv ist, parametriert. Zusätzlich wird bei aktivem Starter (Stc_flgStarterActive = true) oder manuell über das Label Isc_flgIdleSpeedCtrlEnable = false (Standardeinstellung ist true) die Regelung deaktiviert.

Figure 5: Erfassung der Bedingungen für aktiven Leerlaufregler [isc_png_3]

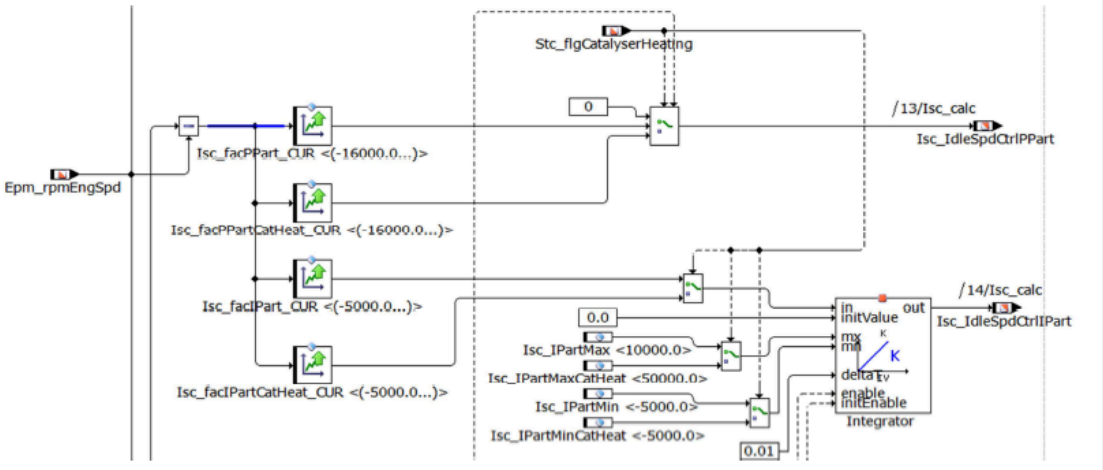


Berechnung der Korrekturgrößen:

Der Leerlaufregler regelt die Drehzahl über 2 Verstellgrößen. Zum einen über die Drosselklappe und zum anderen über die Zündwinkeladaption.

In der Motorsteuerung wird dazu ein Proportional- und ein Integral-Anteil gebildet. Die Drosselklappe ist dabei nur vom I-Anteil1 beeinflusst. Der Zündwinkel dagegen zusätzlich vom schnellen P-Anteil2. Die Verstärkungsfaktoren der beiden Anteile werden in Abhängigkeit der Differenz von Soll-Motordrehzahl und Ist- Motordrehzahl gebildet. Je größer die Differenz ist, desto höher sind die Verstärkungsfaktoren. Parametrierbar sind diese Verstärkungsfaktoren in den Kennfeldern Isc_facPPart_CUR (P-Anteil) und Isc_facIPart_CUR (I-Anteil).

Figure 6: Berechnung P- und I-Anteil des Leerlaufreglers [isc_png_4]



Der P-Anteil (Isc_IdleSpdCtrlPPart) entspricht dem aus Isc_facPPart_CUR gebildeten Verstärkungsfaktor. Beim I-Anteil (Isc_IdleSpdCtrlIPart) wird der Verstärkungsfaktor Isc_facIPart_CUR als Eingang des Integrators verwendet.

Die Grenzen des Integrators werden mit den Parametern Isc_IPartMax und Isc_IPartMin nach oben und unten begrenzt. Delta T des Integrators wird auf 0,01s gesetzt, da diese Funktion im 10ms Raster ausgeführt wird.

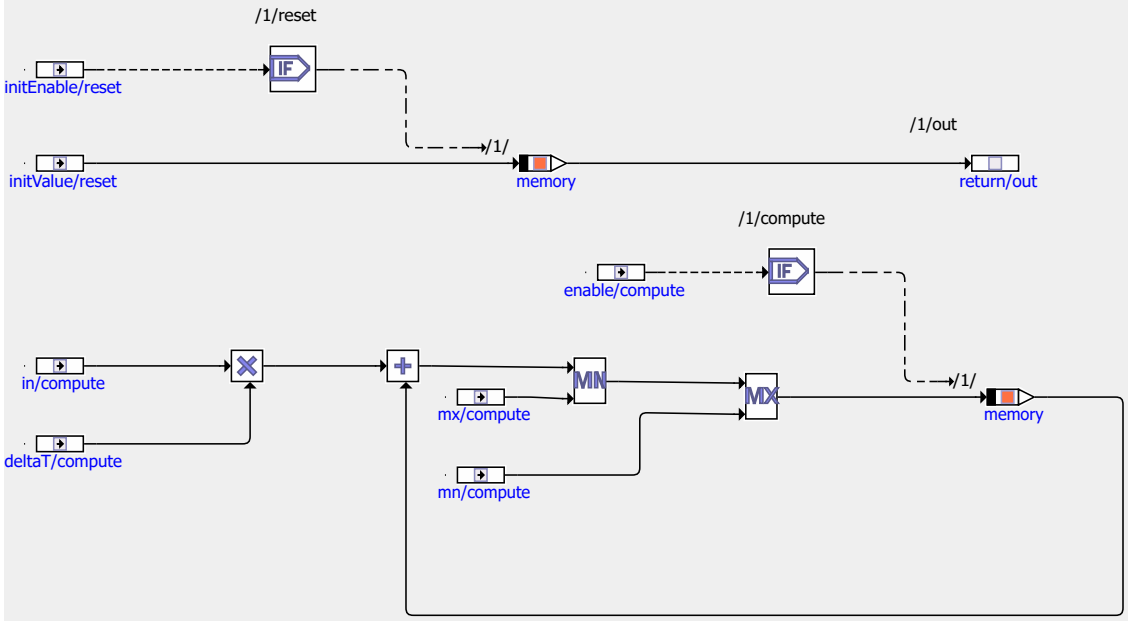
Eine Art Vorsteuerung bildet der Stabilisierungsfaktor Isc_facIdleSpdStab , der auf das Schleppmoment des Motors multipliziert wird. Dieser Faktor wird gebildet aus der Soll-Leerlaufdrehzahl und der Ist-Motordrehzahl und bewirkt eine Anhebung der Momentenanforderung, je niedriger die Motordrehzahl desto höher der Faktor und damit die Momentenanforderung.

Figure 7: [isc_png_5]

$$f_{stab} = \frac{n_{Leerlauf}}{n_{ist}} \times const$$

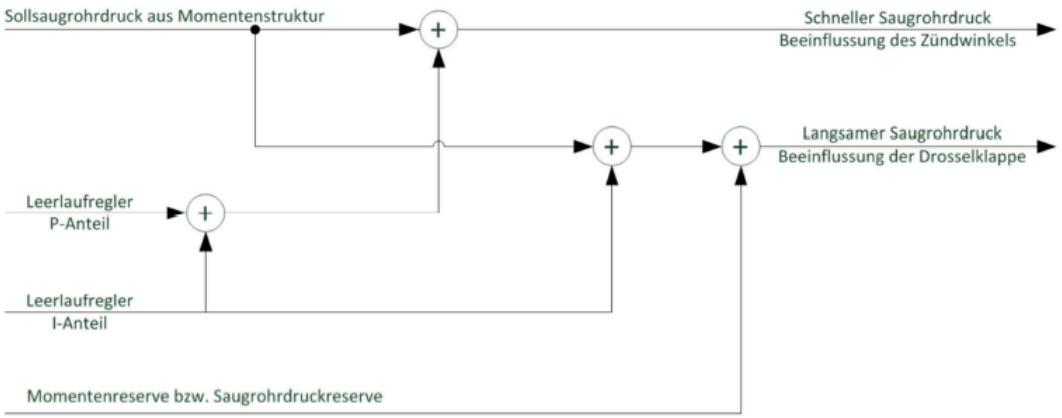
f_{stab}	Stabilitätsfaktor	*Isc_facIdleSpdStab*
$n_{Leerlauf}$	Leerlaufdrehzahl	*Isc_IdleSpdreq*
n_{ist}	Motordrehzahl	*Epm_rpmEngSpd*
const	Korrekturfaktor	*Isc_facEngSpdStab*

Figure 8: Integrator [Integrator.Main]



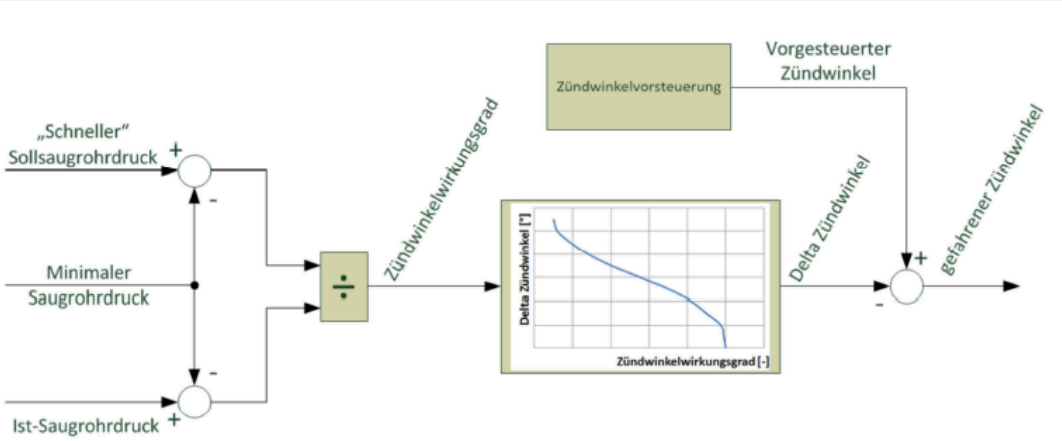
Wird keine Momentenanforderung durch Betätigen des Gaspedals gestellt, sinkt die Drehzahl des Motors ab, solange bis der Leerlaufregler die gewünschte Drehzahl einregelt. Um ein sanfteres Einregelverhalten zu erreichen, wird der Leerlaufregler bereits über der Soll-Leerlaufdrehzahl aktiviert. Die Solldrehzahl, die zum Aktivierungszeitpunkt des Leerlaufreglers der Motordrehzahl entspricht, wird dann über einen Integrator verzögert abgesenkt bis die gewünschte Leerlaufdrehzahl erreicht ist.

Figure 9: Beeinflussung Saugrohrdruck durch den Leerlaufregler [isc_png_6]



Im Folgenden wird erläutert wie der Leerlaufregler funktioniert und auf welche Größen er Einfluss hat. In Abbildung 4-5 ist dargestellt wie der P- und der I-Anteil des Leerlaufreglers auf den Saugrohrdruck einwirken. Aus dem Sollsaugrohrdruck der in der Momentenstruktur berechnet wird, wird ein "schneller“ und ein "langsamer“ Saugrohrdruck gebildet. Auf den „schnellen“ Saugrohrdruck, der auf den Zündwinkel wirkt, werden sowohl P- als auch I-Anteil addiert. Auf den "langsamen" Saugrohrdruck werden der I-Anteil und die Momentenreserve (vorgegeben über T2t_pManiRes) addiert. Dadurch wird der Zündwinkel wesentlich schneller geregelt als die Drosselklappe, was zur Folge hat, dass eine plötzliche Lastmomentänderung über den Zündwinkel ausgeglichen wird bis die Drosselklappe nachfolgt.

Figure 10: Adaption des Zündwinkels durch den Leerlaufregler [isc_png_7]

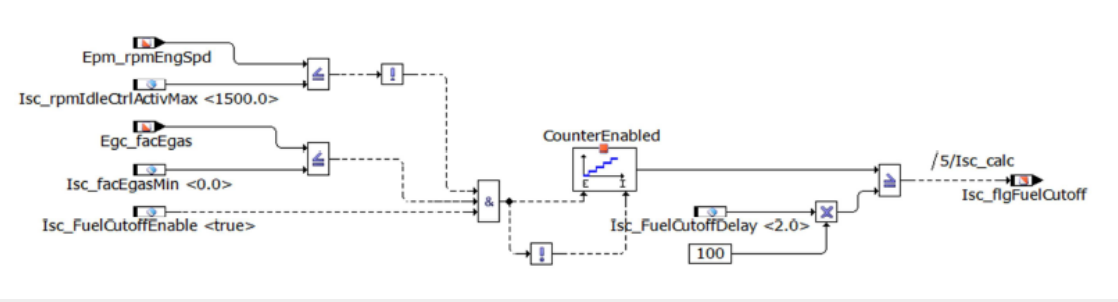


Aus dem "schnellen" Saugrohrdruck und dem Ist-Saugrohrdruck wird in Abbildung 4-6 der minimale Saugrohrdruck, der durch den Saugrohdrucksensor begrenzt wird, subtrahiert und anschließend der Quotient gebildet. Dies ergibt den Zündwinkelwirkungsgrad, der notwendig ist die Drehzahl stabil zu regeln. Durch das Zündwinkelwirkungsgrad-Kennfeld wird aus dem Wirkungsgrad, der Differenz Zündwinkel berechnet und vom vorgesteuerten Zündwinkel abgezogen. Eine Simulation des Leerlaufreglers gegen LABCAR ist in Kapitel 6.1 beschrieben.

1.2.3 Schubbetrieb [Schubbetrieb]

Im Schubbetrieb wird kein abgegebenes Moment des Motors benötigt. Da der Saugrohrdruck nie auf null gehen kann und der Motor mit einem stöchiometrischen Luft-Kraftstoffverhältnis betrieben wird, gibt der Motor trotzdem ein Moment ab. Deshalb soll nach einer bestimmten Zeit, nach Erfüllung der Bedingungen, die Einspritzung abgeschaltet werden. Die Einspritzung wird nicht sofort unterbrochen, da ein zu starkes Ruckeln des Motors vermieden werden soll. Während der aktiven Schubabschaltung ist der Lambdaregler inaktiv geschaltet.

Figure 11: Programmausschnitt Schubbetrieb [isc_png_8]



Bedingungen für die Aktivierung der Schubabschaltung (Abbildung 4-8):

- EGas1-Stellung parametrierbar über Isc_facEgasMin
- Drehzahl über der Schwellendrehzahl des Leerlaufreglers(Isc_rpmIdleCtrlActivMax)
- Schubabschaltung freigegeben (Isc_FuelCutoffEnable = "true")

Die verzögerte Aktivierung der Schubabschaltung erfolgt über einen Counter. Dieser wird aktiv, sobald die Bedingungen erfüllt sind. Ist die Schubabschaltung nicht aktiv, wird der Counter zurückgesetzt. Mit Hilfe des Parameters Isc_FuelCutoffDelay wird die Zeitverzögerung der Schubabschaltung in Sekunden vorgegeben. Ist die Verzögerung erreicht, wird Isc_flgFuelCutoff auf "true" gesetzt.

Sobald der Motor im Schubmodus ist, wird mit Hilfe von Isc_flgFuelCutoff die Lambdaregelung deaktiviert und die Einspritzung unterbrochen.

2 [C-Code Source]

2.1 [Code Listing]

```
/* BEGIN: ASCET REGION "Generation Information" */
/*****************************************************************************
 * BEGIN: Generation Information
 *----------------------------------------------------------------------------
 * Component:.............Class
 * Name:..................."Integrator"
 * Implementation:........."Impl"
 * Dataset:................"Data"
 * Specification:.........Block Diagram
 * Version:...............<empty String>
 * Library Path:..........."smartm160\Project_SmartM160\Lib\Transferfunction"
 *----------------------------------------------------------------------------
 * Project Name:..........."FlexECU_M160"
 * Project Library Path:....."smartm160\Project_SmartM160\"
 *----------------------------------------------------------------------------
 * Generation Date:.......03.12.2014
 * Generation Time:.......13:41:34
 *----------------------------------------------------------------------------
 * ASCET Version:.........V6.1.4 RB-DGS 2.3
 * ASCET-MD Version:......V6.1.4
 * ASCET-RP Version:......V6.1.4
 * ASCET-SE Version:......V6.1.4.28.19 CID[610]
 *----------------------------------------------------------------------------
 * END: Generation Information
 *****************************************************************************/
/* END: ASCET REGION "Generation Information" */

/* BEGIN: ASCET REGION "Project Options" */
/*****************************************************************************
 * BEGIN: Project Options "Build"/"Code"
 *----------------------------------------------------------------------------
 *    Build
 *----------------------------------------------------------------------------
 * Code Generator:.......Object Based Controller Implementation
 * Compiler:.............Microsoft Visual C++ 2008
 * Operating System:.....GENERIC_OSEK
 * Target:...............ANSI-C
 *----------------------------------------------------------------------------
 *    Code
 *----------------------------------------------------------------------------
 * Add Comment with Generation Information for each Component [true]: true
 * Add Comment with Implementation Information for each Assignment Statement [true]: true
 * Add Comment with Specification Source for each Statement [true]:..true
 * Add parenthesis for readability [false]:.....................false
 * Casting [MISRA]:.............................................MISRA
 * Force Parenthesis for Binary Logical Operators [false]:.....false
```

```
 * Generate Define Directives for Enum Values [false]:.............false
 * Prefix for Component Names [<empty String>]:....................<empty String>
 * Protected against division by zero [true]:......................true
 * Protected Division against Signed Overflow [true]:..............true
 * Protected Vector Indices [true]:................................true
 *------------------------------------------------------------------------
 *    Code.Compiler
 *------------------------------------------------------------------------
 * Division truncation direction [Zero (T-division)]:..............Zero (T-division)
 * Inline directive [__inline]:....................................__inline
 * Integer Bit Size [32]:..........................................32
 * Private directive [static]:.....................................static
 * Public directive [<empty String>]:..............................<empty String>
 *------------------------------------------------------------------------
 *    Code.FixedPoint
 *------------------------------------------------------------------------
 * Allow Double bit Size for Division Numerators [true]:.............true
 * Allow Limit Service for Assignment Limitation [true]:.............true
 * Arithmetic Service Set [<None>]:................................<None>
 * Generate Limiters (may be changed locally) [true]:..............true
 * Generate Round Operation on float to integer Assignment [true]:...true
 * Maximum bit Length (float) [64]:................................64
 * Maximum bit Length (int) [32]:..................................32
 * Result on Division by Zero [numerator]:.........................numerator
 * Temp Vars always 32 bit (integer) [false]:......................false
 * Use power of 2 approximations of literals [false]:..............false
 * Use SHIFT Operation on Signed Values instead of DIV Operation [true]: true
 * Use SHIFT Operation on Signed Values instead of MUL Operation [true]: true
 *------------------------------------------------------------------------
 *    Code.Optimizations
 *------------------------------------------------------------------------
 * Auto-inline private methods (Smaller code-size - may be changed locally) [false]: false
 * Generate well-formed switch [false]:............................false
 * Hierarchical Code-Generation (may be changed locally) [false]:....false
 * Initialise history variable with zero [false]:..................false
 * Optimize Direct Access Methods (Multiple Levels) [false]:.......false
 * Optimize Direct Access Methods (One Level) [false]:.............false
 * Optimize Static Actions (Restricted Modelling) [false]:.........false
 * Outline Generated Methods (may be changed locally) [false]:.....false
 *------------------------------------------------------------------------
 *    Code.Production
 *------------------------------------------------------------------------
 * Add Implementation Definitions [true]:..........................true
 * Generate Access Macros for [(variables, messages)]:.............(variables, messages)
 * Generate Access Methods for dT (Alternative: use OS dT directly) [true]: true
 * Generate Data Structures [USELOCAL]:............................USELOCAL
 * Generate Map File [true]:.......................................true
 * Generate OS Configuration [true]:...............................true
 *------------------------------------------------------------------------
 *    Station.Build
 *------------------------------------------------------------------------
 * Use Customized Data Type Names [false]:.........................false
 *------------------------------------------------------------------------
 * END: Project Options "Build"/"Code"
 **************************************************************************/
/* END: ASCET REGION "Project Options" */
/* BEGIN: ASCET REGION "ASCET-SE AddOn Options" */
/**************************************************************************
 * BEGIN: ASCET-SE AddOn Options
 * ------------------------------------------------------------------------
 *     Code
 * ------------------------------------------------------------------------
 * checkMemSectionVolatility [true]:...............................false
 * checkMultipleSend [false]:......................................false
 * distribVarMemClass ["DISTRRAM"]:................................"RAM"
 * genAlwaysInitValues [false]:....................................true
 * genLogicElementsAs [PACKED_BITFIELD]:...........................PACKED_BITFIELD
 * genObjList [false]:.............................................false
 * implInfoComments [true]:........................................true
 * initTaskMemClass ["ASD_INIT_TASK_MEM"]:.........................."ASD_INIT_TASK_MEM"
 * isrMemClass ["ASD_ISR_MEM"]:....................................."ASD_ISR_MEM"
 * mainMemClass ["ASD_EXT_CODE_MEM"]:..............................."ASD_EXT_CODE_MEM"
 * optimizeUnusedCode [true]:......................................true
 * paramAsSysConst [false]:........................................false
 * pragmaMemClassAtDecl [false]:...................................false
 * pragmaMemClassEnabled [true]:...................................false
 * referenceMemClass ["REFRAM"]:...................................."RAM"
 * shortNames [false]:.............................................false
 * taskMemClass ["ASD_TASK_MEM"]:..................................."ASD_TASK_MEM"
 * virtualParameterMemClass ["VIRT_PARAM"]:........................."VIRT_PARAM"
 * ------------------------------------------------------------------------
 *     Code.Appearance
 * ------------------------------------------------------------------------
 * braceLineFeed [true]:...........................................true
 * genDate [<undef>]:..............................................<undef>
```

```
 * genTime [<undef>]:.....<undef>
 * generateSignatureDecorationComments [true]:.....true
 * lineFeedPosition [LEFT]:.....LEFT
 * maxIdentLength [0]:.....40
 * maxRightLength [60]:.....60
 * minLeftLength [8]:.....8
 * preventIndentStructInit [true]:.....true
 * ------------------------------------------------------------------------
 *     OS
 * ------------------------------------------------------------------------
 * Os-Config-C_gen_declaration_alarms [false]:.....false
 * Os-Config-C_gen_declaration_appmodes [false]:.....false
 * Os-Config-C_gen_dt_calc [false]:.....true
 * Os-Config-C_gen_initCOM [false]:.....false
 * Os-Config-C_gen_inittask [false]:.....true
 * Os-Config-C_gen_main [false]:.....false
 * Os-Config-C_gen_process_container [false]:.....true
 * Os-Config-C_gen_startuphook [false]:.....false
 * asd_exclusive_area ["ASD_EXCLUSIVE_AREA"]:......"ASCET_exclusive_area"
 * messageDoInit [false]:.....false
 * messageExternalMessageCopies [false]:.....false
 * messageGenOSEKDeclarations [true]:.....false
 * messageIgnoreUsageInInitTask [false]:.....false
 * messageOverloadInitValues [<undef>]:.....<undef>
 * messageUsageVariant [OPT_COPY]:.....NON_OPT_COPY
 * modularMessageUse [false]:.....false
 * osAppModePattern ["%name%"]:....."appmode_%name%"
 * osStartupFunction [<undef>]:.....<undef>
 * ------------------------------------------------------------------------
 *     OS.OIL
 * ------------------------------------------------------------------------
 * OIL-COOP-RESOURCE-name ["ASD_Cooperative_Res"]:....."ASD_Cooperative_Res"
 * OIL-outputFile ["temp.oil"]:....."temp.oil"
 * ------------------------------------------------------------------------
 *     SERAP
 * ------------------------------------------------------------------------
 * SERAPRefPageMemoryClass ["SERAP_REF"]:....."SERAP_REF"
 * SERAPWorkPageMemoryClass ["SERAP_WORK"]:....."SERAP_WORK"
 * serap [false]:.....false
 * serapEmbedded [true]:.....true
 * ------------------------------------------------------------------------
 *     Virtual Address Tables
 * ------------------------------------------------------------------------
 * addressTable [true]:.....false
 * addressTableMemoryClass ["VATROM"]:....."VATROM"
 * ------------------------------------------------------------------------
 * END: ASCET-SE AddOn Options
 ******************************************************************************/

/* END: ASCET REGION "ASCET-SE AddOn Options" */
/*****************************************************************************
 * BEGIN: source code of a multiple instance class
 ******************************************************************************/

/* BEGIN: ASCET REGION "Exported Data Definitions" */
/* END: ASCET REGION "Exported Data Definitions" */
#define _memory self->INTEGRATOR_IMPL_RAM->memory

/* BEGIN: ASCET REGION "Component Functions" */
/*****************************************************************************
 * BEGIN: FUNCTIONS OF COMPONENT
 ******************************************************************************/

/* BEGIN: ASCET REGION "Method Definition 'compute'" */
/*****************************************************************************
 * BEGIN: DEFINITION OF METHOD 'INTEGRATOR_IMPL_compute'
 * ------------------------------------------------------------------------
 * model name:.....'compute'
 * memory class:.....'CODE'
 * ------------------------------------------------------------------------*/
//#if defined(COMPILE_UNUSED_CODE) || defined(COMPILE_UNUSED__INTEGRATOR_IMPL_compute)
/* public compute (in::cont;enable::log;deltaT::cont;mx::cont;mn::cont)  */

void INTEGRATOR_IMPL_compute (
              const struct INTEGRATOR_IMPL * self,
   /* IN    */ real64                       in,
   /* IN    */ uint8                        enable,
   /* IN    */ real64                       deltaT,
   /* IN    */ real64                       mx,
   /* IN    */ real64                       mn
   )
{
   /* temp. variables */
   real64 _t1real64;
```

```
   /* compute: sequence call #1  */
   if (enable)
   {
      /* If-block: sequence call #1/Then #1  */
      _t1real64 = in * deltaT + _memory;
      _t1real64 = ((_t1real64 <= mx) ? _t1real64 : mx);
      /* assignment to memory: min=-oo, max=+oo, hex=phys, limit=n.a., zero incl.=true  */
      _memory = ((_t1real64 >= mn) ? _t1real64 : mn);
   } /* end if */
}
/* ---------------------------------------------------------------------------
 * END: DEFINITION OF METHOD 'INTEGRATOR_IMPL_compute'
 ***************************************************************************/
#endif
/* END: ASCET REGION "Method Definition 'compute'" */


/* BEGIN: ASCET REGION "Method Definition 'out'" */
/****************************************************************************
 * BEGIN: DEFINITION OF METHOD 'INTEGRATOR_IMPL_out'
 * -------------------------------------------------------------------------
 * model name:.................................'out'
 * memory class:...............................'CODE'
 * -------------------------------------------------------------------------*/
//#if defined(COMPILE_UNUSED_CODE) || defined(COMPILE_UNUSED__INTEGRATOR_IMPL_out)
/* public out () return::cont  */

real64 INTEGRATOR_IMPL_out ( const struct INTEGRATOR_IMPL * self)
{
   /* out: sequence call #1  */
   /* return with expr from out: min=-oo, max=+oo, hex=phys, limit=n.a., zero incl.=true  */
   return _memory;
}
/* ---------------------------------------------------------------------------
 * END: DEFINITION OF METHOD 'INTEGRATOR_IMPL_out'
 ***************************************************************************/
#endif
/* END: ASCET REGION "Method Definition 'out'" */


/* BEGIN: ASCET REGION "Method Definition 'reset'" */
/****************************************************************************
 * BEGIN: DEFINITION OF METHOD 'INTEGRATOR_IMPL_reset'
 * -------------------------------------------------------------------------
 * model name:.................................'reset'
 * memory class:...............................'CODE'
 * -------------------------------------------------------------------------*/
//#if defined(COMPILE_UNUSED_CODE) || defined(COMPILE_UNUSED__INTEGRATOR_IMPL_reset)
/* public reset (initValue::cont;initEnable::log)  */

void INTEGRATOR_IMPL_reset (
              const struct INTEGRATOR_IMPL * self,
   /* IN    */ real64                       initValue,
   /* IN    */ uint8                        initEnable
   )
{
   /* reset: sequence call #1  */
   if (initEnable)
   {
      /* If-block: sequence call #1/Then #1  */
      /* assignment to memory: min=-oo, max=+oo, hex=phys, limit=n.a., zero incl.=true  */
      _memory = initValue;
   } /* end if */
}
/* ---------------------------------------------------------------------------
 * END: DEFINITION OF METHOD 'INTEGRATOR_IMPL_reset'
 ***************************************************************************/
#endif
/* END: ASCET REGION "Method Definition 'reset'" */


/* ***************************************************************************
 * END: FUNCTIONS OF COMPONENT
 ***************************************************************************/
/* END: ASCET REGION "Component Functions" */
```