Amf - Air Mass Flow

1 [Air Mass Flow] Air Mass Flow

1.1 [Overview]

Figure 1: [Amf Function Overview]

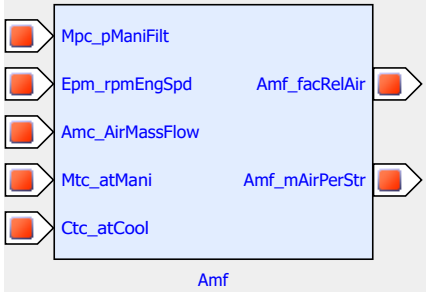
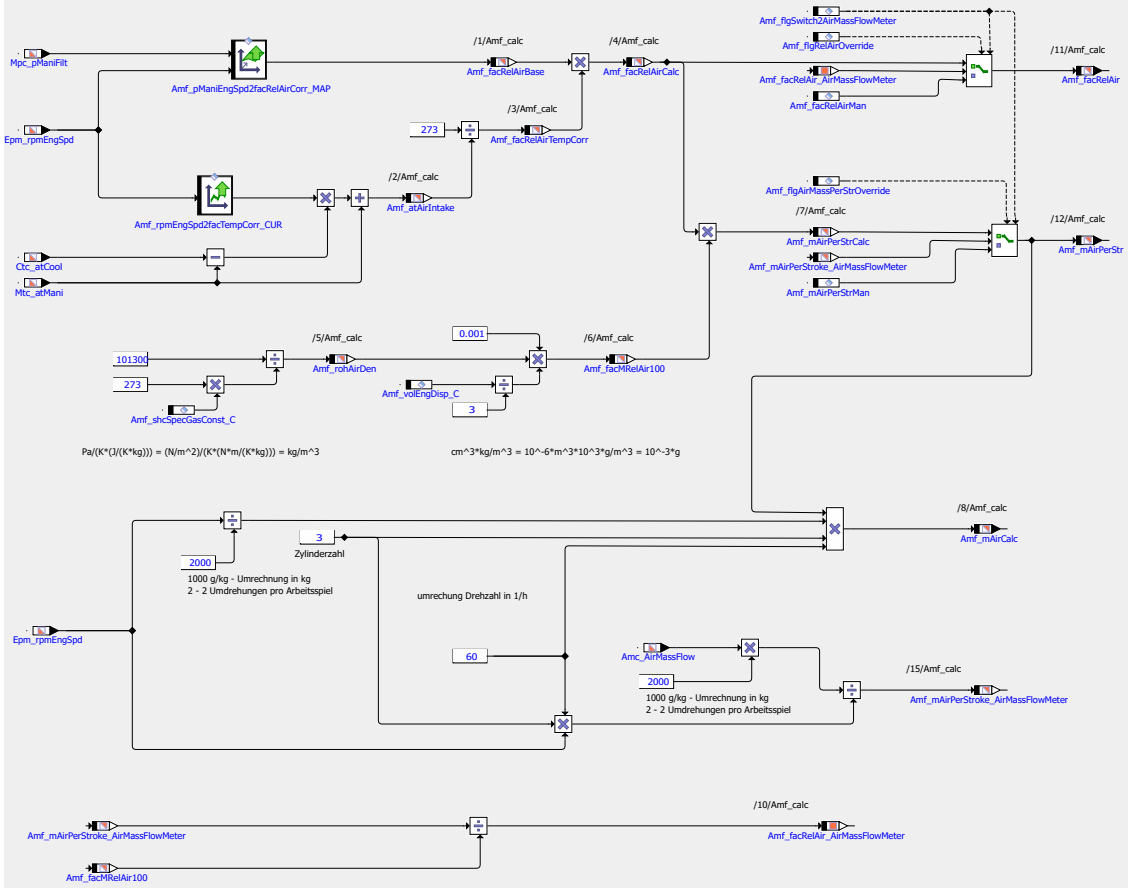


Figure 2: [AirMassFlow.Main]



1.2 Luftmassenstrom [Luftmassenstrom]

Im Modul Luftmassenstrom (AirMassFlow) wird aus dem Saugrohrdruck Mpc_pManiFilt und der Drehzahl Epm_rpmEngSpd die relative Füllung Amf_facRelAir und die Luftmasse pro Absaugung Amf_mAirPerStr berechnet und als Nachricht zur Verfügung gestellt. Die relative Füllung ist die Frischluftmasse im Zylinder unter den dort vorherrschenden Bedingungen relativ zur Frischluftmasse im selben Volumen unter Normbedingungen (Druck von 1013hPa, Temperatur von 273K). Die relative Füllung ist bei Saugmotoren linear vom Saugrohrdruck abhängig, wobei der Restgaspartialdruck berücksichtigt werden muss, da sich der Zylinder beim Auslassen des verbrannten Gemisches aufgrund des Totvolumens nicht vollständig entleert. Gleichung 5.3 beschreibt den Zusammenhang zwischen Saugrohrdruck und relativer Füllung, der in der Kennlinie Amf_pManiEngSpd2facRelAirCorr_MAP hinterlegt

ist. Als Grundwert für den Restgaspartialdruck wird 150hPa verwendet. Die Steigung errechnet sich aus dem zweiten Punkt mit 90% bei 1013hPa. Beide Werte entspringen Erfahrungen aus existierenden Motorsteuerungen.

Figure 3: [amf_png_1]

$$rl = (ps - pbrint) \times fupsrl$$

rl	relative Füllung `Amf_facRelAirBase`
ps	Saugrohrdruck `Mpc_pManiFilt`
$pbrint$	Restgaspartialdruck
$fupsrl$	Faktor zur Umrechnung Saugrohrdruck in relative Füllung

Das es sich bei dem Smartmotor um einen mit einem Abgasturbolader aufgeladenen Motor handelt muss die Kennlinie noch verifiziert und gegebenenfalls angepasst werden. Über die Kennlinie Amf_rpmEngSpd2facTempCorr_CUR kann bei Bedarf noch eine drehzahlabhängige Korrektur durchgeführt werden. Die Kennlinie ist als Grundwert durchgehend mit Eins gefüllt.

Um den Einfluss der Temperatur zu Berücksichtigen wird der berechnete Basiswert der relativen Füllung mit dem Faktor Amf_facRelAirTempCorr beaufschlagt, der das Verhältnis von Normtemperatur zur Temperatur am Einlass darstellt. Die Temperatur im Zylinder bei Frischluftansaugung wird nicht gemessen und muss daher berechnet werden. Beim Durchströmen des Einlassventils und dem Verteilen im Zylinder erwärmt sich die Luft am Ventil selbst, sowie an den Zylinderwänden, wobei die Drehzahl eine entscheidende Rolle spielt. Sie bestimmt die Zeit, die die Luft hat, um sich zu erwärmen. Zur Abschätzung der Temperatur der Frischluft im Zylinder wird die Differenztemperatur zwischen Kühlmittel Ctc_atCool (quasi Temperatur des Motorblocks) und der Frischluft Mtc_atMani gebildet, mit dem drehzahlabhängigen Faktor Amf_rpmEngSpd2facTempCorr_CUR multipliziert und anschließend die Frischlufttemperatur addiert. Gleichung 5.4 beschreibt den Zusammenhang zur Abschätzung der Frischlufttemperatur Amf_atAirIntake bei Einlass.

Figure 4: [amf_png_2]

$$T_{Einlass} = (T_{Kühlmittel} - T_{Luft}) \times F_{Korrektur} + T_{Luft}$$

Um die abgesaugte Luftmasse pro Hub zu Berechnen wird die Luftmasse unter Normbedingungen berechnet und mit der relativen Füllung multipliziert. Gleichung 5.5 beschreibt den Zusammenhang zur Berechnung der Luftdichte Amf_rohAirDen und daraus der Luftmasse für den Smartmotor.

Figure 5: [amf_png_3]

$$\rho = \frac{p}{R_S \times T}$$

ρ	Dichte
p	Druck
R_S	spezifische Gaskonstante
T	Temperatur

$$m = \rho \times V$$

ρ	Dichte
m	Masse
V	Volumen

Da es sich um in der Regel Konstante Werte handelt, könnte die Normluftmasse auch direkt als Parameter eingehen. Durch die hier gewählte Implementierung zur Berechnung der Normluftmasse ist jedoch die Möglichkeit für schnelle Änderungen geben.

Durch setzen des Flags Amf_flgRelAirOverride kann die relative Füllung für die weiteren Berechnungen manuell durch Amf_facRelAirMan überschrieben werden und durch das Flag Amf_flgAirMassPerStrOverride die Luftmasse pro Absaugung eines Zylinders mit Amf_mAirPerStrMan . Die gesamte Auswertung erfolgt durch die Funktion Amf_calc im Synchron-Raster.

Die Idee, die Luftmassenerfassung in dieser Art umzusetzen, entstammt dem Skript "Engine Control Systems" von Georg Mallebrein.

2 [C-Code Source]

2.1 [Code Listing]

```
/* BEGIN: ASCET REGION "Generation Information" */
/*****************************************************************************
 * BEGIN: Generation Information
 *----------------------------------------------------------------------------
 * Component:.............Module
 * Name:..................."AirMassFlow"
 * Implementation:........."Impl"
 * Dataset:................"Data"
 * Specification:..........Block Diagram
 * Version:...............<empty String>
 * Library Path:..........."smartm160\Project_SmartM160\Function_Modules"
 *----------------------------------------------------------------------------
 * 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: ASCET REGION "Module Data Definitions" */

/******************************************************************************
 * BEGIN: DEFINITION OF SUBSTRUCT VARIABLE 'Amf_RAM'
 * ---------------------------------------------------------------------------
 * memory class:..............................'RAM'
 * model name:................................'Amf'
 * data set:..................................'AIRMASSFLOW_IMPL_Data'
 * ---------------------------------------------------------------------------*/
struct AIRMASSFLOW_IMPL_RAM_SUBSTRUCT Amf_RAM = {
   /* struct element:'Amf_RAM.Amf_facRelAir_AirMassFlowMeter' (modeled as:'Amf_facRelAir_AirMassFlowMeter.Amf')
 */
   0
};
/* ---------------------------------------------------------------------------
 * END: DEFINITION OF SUBSTRUCT VARIABLE 'Amf_RAM'
 ******************************************************************************/

/******************************************************************************
 * DEFINITION OF COMPONENT VARIABLE OMITTED
 * ---------------------------------------------------------------------------
 * memory class:..............................'ROM'
 * model name:................................'Amf'
 * reason:....................................no local elements
 * ---------------------------------------------------------------------------*/

/* END: ASCET REGION "Module Data Definitions" */

/* BEGIN: ASCET REGION "Exported Data Definitions" */

/******************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_atAirIntake'
 * ---------------------------------------------------------------------------*/
uint16 Amf_atAirIntake = 128;
   /* min=0.0078125, max=511.9921875, fac_128, limit=yes */
/* ---------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_atAirIntake'
 ******************************************************************************/


/******************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_facMRelAir100'
 * ---------------------------------------------------------------------------*/
uint16 Amf_facMRelAir100 = 0;
   /* min=0.0, max=4.0, fac_10000, limit=yes */
/* ---------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_facMRelAir100'
 ******************************************************************************/


/******************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_facRelAirBase'
 * ---------------------------------------------------------------------------*/
```

```
uint16 Amf_facRelAirBase = 0;
   /* min=0.0, max=2.5, fac_512, limit=yes */
/* ----------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_facRelAirBase'
 ****************************************************************************/


/****************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_facRelAirCalc'
 * --------------------------------------------------------------------------*/
uint16 Amf_facRelAirCalc = 0;
   /* min=0.0, max=2.5, fac_512, limit=yes */
/* ----------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_facRelAirCalc'
 ****************************************************************************/


/****************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_facRelAirMan'
 * --------------------------------------------------------------------------*/
const uint16 Amf_facRelAirMan = 0;
   /* min=0.0, max=2.5, fac_512, limit=yes */
/* ----------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_facRelAirMan'
 ****************************************************************************/


/****************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_facRelAirTempCorr'
 * --------------------------------------------------------------------------*/
uint8 Amf_facRelAirTempCorr = 0;
   /* min=0.0, max=1.9921875, fac_128, limit=yes */
/* ----------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_facRelAirTempCorr'
 ****************************************************************************/


/****************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_flgAirMassPerStrOverride'
 * --------------------------------------------------------------------------*/
const uint8 Amf_flgAirMassPerStrOverride = false;
   /* min=0, max=1, Identity, limit=yes */
/* ----------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_flgAirMassPerStrOverride'
 ****************************************************************************/


/****************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_flgRelAirOverride'
 * --------------------------------------------------------------------------*/
const uint8 Amf_flgRelAirOverride = false;
   /* min=0, max=1, Identity, limit=yes */
/* ----------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_flgRelAirOverride'
 ****************************************************************************/


/****************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_flgSwitch2AirMassFlowMeter'
 * --------------------------------------------------------------------------*/
const uint8 Amf_flgSwitch2AirMassFlowMeter = false;
   /* min=0, max=1, Identity, limit=yes */
/* ----------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_flgSwitch2AirMassFlowMeter'
 ****************************************************************************/


/****************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_mAirPerStrCalc'
 * --------------------------------------------------------------------------*/
uint16 Amf_mAirPerStrCalc = 0;
   /* min=0.0, max=9.99984741210938e-1, fac_65536, limit=yes */
/* ----------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_mAirPerStrCalc'
 ****************************************************************************/


/****************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_mAirPerStrMan'
 * --------------------------------------------------------------------------*/
const uint16 Amf_mAirPerStrMan = 0;
   /* min=0.0, max=9.99984741210938e-1, fac_65536, limit=yes */
/* ----------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_mAirPerStrMan'
 ****************************************************************************/
```

```
/**********************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_mAirPerStroke_AirMassFlowMeter'
 * --------------------------------------------------------------------------------*/
uint16 Amf_mAirPerStroke_AirMassFlowMeter = 0;
   /* min=0.0, max=9.99984741210938e-1, fac_65536, limit=yes */
/* --------------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_mAirPerStroke_AirMassFlowMeter'
 **********************************************************************************/


/**********************************************************************************
 * BEGIN: DEFINITION OF CHARACTERISTIC TABLE 'Amf_pManiEngSpd2facRelAirCorr_MAP'
 * --------------------------------------------------------------------------------*/
const struct CharTable2_uint16_10_sint16_14_uint16_AIRMASSFLOW_IMPL_TYPE Amf_pManiEngSpd2facRelAirCorr_MAP = {
      10,
      14,
      {
         7500, 12500, 17500, 22500, 27500, 32500, 37500, 42500, 47500, 52500
      },
      {
         1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000, 11000, 12000,
         13000, 14000
      },
      {
         148, 148, 146, 149, 149, 150, 145, 147, 146, 146, 151, 152, 152, 153,
         224, 224, 223, 230, 227, 232, 227, 224, 224, 217, 223, 223, 223, 223,
         317, 316, 312, 312, 312, 316, 316, 312, 307, 301, 306, 305, 303, 302,
         414, 412, 409, 409, 404, 409, 409, 404, 404, 390, 396, 394, 392, 390,
         521, 518, 521, 505, 505, 505, 511, 501, 501, 491, 492, 489, 486, 484,
         629, 625, 618, 627, 608, 608, 613, 603, 603, 594, 594, 590, 586, 582,
         734, 731, 720, 716, 734, 715, 715, 711, 711, 691, 698, 694, 691, 687,
         839, 834, 822, 817, 842, 827, 812, 808, 812, 793, 798, 794, 790, 787,
         939, 934, 923, 918, 933, 914, 909, 903, 905, 885, 888, 884, 879, 873,
         1042, 1036, 1025, 1017, 1036, 1014, 1008, 1000, 1005, 982, 985, 979, 974, 968
      }
   };
   /* result: min=0.0, max=2.5, fac_512, limit=no */
   /* x axis: min=0.0, max=262140.0, fac_1div4 */
   /* y axis: min=-16384.0, max=16383.5, fac_2 */
/* --------------------------------------------------------------------------------
 * END: DEFINITION OF CHARACTERISTIC TABLE 'Amf_pManiEngSpd2facRelAirCorr_MAP'
 **********************************************************************************/


/**********************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_rohAirDen'
 * --------------------------------------------------------------------------------*/
uint16 Amf_rohAirDen = 0;
   /* min=0.0, max=2.0, fac_4096, limit=yes */
/* --------------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_rohAirDen'
 **********************************************************************************/


/**********************************************************************************
 * BEGIN: DEFINITION OF CHARACTERISTIC TABLE 'Amf_rpmEngSpd2facTempCorr_CUR'
 * --------------------------------------------------------------------------------*/
const struct CharTable1_sint16_19_uint8_AIRMASSFLOW_IMPL_TYPE Amf_rpmEngSpd2facTempCorr_CUR = {
      19,
      {
         1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000, 5500, 6000, 7000, 8000,
         9000, 10000, 11000, 12000, 13000, 14000
      },
      {
         127, 92, 78, 70, 65, 63, 60, 59, 58, 56, 55, 54, 52, 52, 51, 51, 51, 51, 51
      }
   };
   /* result: min=0.0, max=0.99609375, fac_256, limit=no */
   /* x axis: min=0.0, max=16383.5, fac_2 */
/* --------------------------------------------------------------------------------
 * END: DEFINITION OF CHARACTERISTIC TABLE 'Amf_rpmEngSpd2facTempCorr_CUR'
 **********************************************************************************/


/**********************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_shcSpecGasConst_C'
 * --------------------------------------------------------------------------------*/
const uint16 Amf_shcSpecGasConst_C = 2296;
   /* min=0.125, max=375.0, fac_8, limit=yes */
/* --------------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_shcSpecGasConst_C'
 **********************************************************************************/
```

```
/*****************************************************************************
 * BEGIN: DEFINITION OF VARIABLE 'Amf_volEngDisp_C'
 * ---------------------------------------------------------------------------*/
const uint16 Amf_volEngDisp_C = 699;
   /* min=1.0, max=10000.0, ident, limit=yes */
/* ---------------------------------------------------------------------------
 * END: DEFINITION OF VARIABLE 'Amf_volEngDisp_C'
 *****************************************************************************/

/* END: ASCET REGION "Exported Data Definitions" */

/*****************************************************************************
 * BEGIN: DEFINITION OF MESSAGES
 * ---------------------------------------------------------------------------
 * Total size is [bytes]:.....................6
 * ---------------------------------------------------------------------------*/
/* messages of memory class:.....................................'RAM' */
/* messages of size [bytes]:.....................................2 */
/* modelled as 'Amf_facRelAir' */
uint16 Amf_facRelAir;
/* modelled as 'Amf_mAirCalc' */
uint16 Amf_mAirCalc;
/* modelled as 'Amf_mAirPerStr' */
uint16 Amf_mAirPerStr;
/* ---------------------------------------------------------------------------
 * END: DEFINITION OF MESSAGES
 *****************************************************************************/

#define _Amf_atAirIntake Amf_atAirIntake
#define _Amf_facMRelAir100 Amf_facMRelAir100
#define _Amf_facRelAir_AirMassFlowMeter Amf_RAM.Amf_facRelAir_AirMassFlowMeter
#define _Amf_facRelAirBase Amf_facRelAirBase
#define _Amf_facRelAirCalc Amf_facRelAirCalc
#define _Amf_facRelAirMan Amf_facRelAirMan
#define _Amf_facRelAirTempCorr Amf_facRelAirTempCorr
#define _Amf_flgAirMassPerStrOverride Amf_flgAirMassPerStrOverride
#define _Amf_flgRelAirOverride Amf_flgRelAirOverride
#define _Amf_flgSwitch2AirMassFlowMeter Amf_flgSwitch2AirMassFlowMeter
#define _Amf_mAirPerStrCalc Amf_mAirPerStrCalc
#define _Amf_mAirPerStrMan Amf_mAirPerStrMan
#define _Amf_mAirPerStroke_AirMassFlowMeter Amf_mAirPerStroke_AirMassFlowMeter
#define _Amf_pManiEngSpd2facRelAirCorr_MAP Amf_pManiEngSpd2facRelAirCorr_MAP
#define _Amf_pManiEngSpd2facRelAirCorr_MAP_REF_ (&(Amf_pManiEngSpd2facRelAirCorr_MAP))
#define _Amf_rohAirDen Amf_rohAirDen
#define _Amf_rpmEngSpd2facTempCorr_CUR Amf_rpmEngSpd2facTempCorr_CUR
#define _Amf_rpmEngSpd2facTempCorr_CUR_REF_ (&(Amf_rpmEngSpd2facTempCorr_CUR))
#define _Amf_shcSpecGasConst_C Amf_shcSpecGasConst_C
#define _Amf_volEngDisp_C Amf_volEngDisp_C

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

/* BEGIN: ASCET REGION "Process Definition 'Amf_calc'" */
/*****************************************************************************
 * BEGIN: DEFINITION OF PROCESS 'AIRMASSFLOW_IMPL_Amf_calc'
 * ---------------------------------------------------------------------------
 * model name:.................................'Amf_calc'
 * memory class:...............................'CODE'
 * ---------------------------------------------------------------------------*/
//#if defined(COMPILE_UNUSED_CODE) || defined(COMPILE_UNUSED__AIRMASSFLOW_IMPL_Amf_calc)
/* messages used by this process */

/* public Amf_calc []  */

void AIRMASSFLOW_IMPL_Amf_calc (void)
{
   /* temp. variables */
   sint32 _t1sint32;
   uint32 _t1uint32;
   uint16 _t1uint16;
   uint32 _t2uint32;

   /* define local message copies */
   uint16 Amc_AirMassFlow__AIRMASSFLOW_IMPL_Amf_calc;
   uint16 Amf_facRelAir__AIRMASSFLOW_IMPL_Amf_calc;
   uint16 Amf_mAirCalc__AIRMASSFLOW_IMPL_Amf_calc;
   uint16 Amf_mAirPerStr__AIRMASSFLOW_IMPL_Amf_calc;
   uint16 Ctc_atCool__AIRMASSFLOW_IMPL_Amf_calc;
   sint16 Epm_rpmEngSpd__AIRMASSFLOW_IMPL_Amf_calc;
   uint16 Mpc_pManiFilt__AIRMASSFLOW_IMPL_Amf_calc;
   uint16 Mtc_atMani__AIRMASSFLOW_IMPL_Amf_calc;
   /* receive messages implicitly */
```

```
  {
     DisableAllInterrupts();
     Amc_AirMassFlow__AIRMASSFLOW_IMPL_Amf_calc = Amc_AirMassFlow;
     Amf_facRelAir__AIRMASSFLOW_IMPL_Amf_calc = Amf_facRelAir;
     Amf_mAirCalc__AIRMASSFLOW_IMPL_Amf_calc = Amf_mAirCalc;
     Amf_mAirPerStr__AIRMASSFLOW_IMPL_Amf_calc = Amf_mAirPerStr;
     Ctc_atCool__AIRMASSFLOW_IMPL_Amf_calc = Ctc_atCool;
     Epm_rpmEngSpd__AIRMASSFLOW_IMPL_Amf_calc = Epm_rpmEngSpd;
     Mpc_pManiFilt__AIRMASSFLOW_IMPL_Amf_calc = Mpc_pManiFilt;
     Mtc_atMani__AIRMASSFLOW_IMPL_Amf_calc = Mtc_atMani;
     EnableAllInterrupts();
  }
  /* Amf_calc: sequence call #1  */
  /* assignment to Amf_facRelAirBase: min=0, max=1280, hex=512phys+0, limit=(maxBitLength: true, assign: true),
zero incl.=true  */
  _Amf_facRelAirBase
     =
CharTable2_getAt_u16s16u16(_Amf_pManiEngSpd2facRelAirCorr_MAP_REF_,Mpc_pManiFilt__AIRMASSFLOW_IMPL_Amf_calc,Epm_rpmEn
  /* Amf_calc: sequence call #2  */
  _t1sint32
     =
((sint32)CharTable1_getAt_s16u8(_Amf_rpmEngSpd2facTempCorr_CUR_REF_,Epm_rpmEngSpd__AIRMASSFLOW_IMPL_Amf_calc)
* ((sint32)Ctc_atCool__AIRMASSFLOW_IMPL_Amf_calc - (sint32)Mtc_atMani__AIRMASSFLOW_IMPL_Amf_calc) >> 8) +
(sint32)Mtc_atMani__AIRMASSFLOW_IMPL_Amf_calc;
  _t1sint32
     = ((_t1sint32 >= 1) ? ((_t1sint32 <= 65535) ? _t1sint32 : 65535) : 1);
  /* assignment to Amf_atAirIntake: min=1, max=65535, hex=128phys+0, limit=(maxBitLength: true, assign: true),
zero incl.=true  */
  _Amf_atAirIntake = (uint16)_t1sint32;
  /* Amf_calc: sequence call #3  */
  _t1uint32 = 4472832U / _Amf_atAirIntake;
  /* assignment to Amf_facRelAirTempCorr: min=0, max=255, hex=128phys+0, limit=(maxBitLength: true, assign:
true), zero incl.=true  */
  _Amf_facRelAirTempCorr = (uint8)(((_t1uint32 <= 255U) ? _t1uint32 : 255U));
  /* Amf_calc: sequence call #4  */
  _t1uint32 = (uint32)_Amf_facRelAirBase * _Amf_facRelAirTempCorr;
  /* assignment to Amf_facRelAirCalc: min=0, max=1280, hex=512phys+0, limit=(maxBitLength: true, assign: true),
zero incl.=true  */
  _Amf_facRelAirCalc = (uint16)(((_t1uint32 <= 163840U) ? _t1uint32 >> 7 : 1280U));
  /* Amf_calc: sequence call #5  */
  _t1uint32 = 12158968U / _Amf_shcSpecGasConst_C;
  /* assignment to Amf_rohAirDen: min=0, max=8192, hex=4096phys+0, limit=(maxBitLength: true, assign: true),
zero incl.=true  */
  _Amf_rohAirDen = (uint16)(((_t1uint32 <= 8192U) ? _t1uint32 : 8192U));
  /* Amf_calc: sequence call #6  */
  _t1uint32 = (uint32)_Amf_rohAirDen * _Amf_volEngDisp_C * 5U / 6144U;
  /* assignment to Amf_facMRelAir100: min=0, max=40000, hex=10000phys+0, limit=(maxBitLength: true, assign:
true), zero incl.=true  */
  _Amf_facMRelAir100 = (uint16)(((_t1uint32 <= 40000U) ? _t1uint32 : 40000U));
  /* Amf_calc: sequence call #7  */
  _t1uint32
     = ((uint32)_Amf_facRelAirCalc * _Amf_facMRelAir100 <<3) / 625U;
  /* assignment to Amf_mAirPerStrCalc: min=0, max=65535, hex=65536phys+0, limit=(maxBitLength: true, assign:
true), zero incl.=true  */
  _Amf_mAirPerStrCalc = (uint16)(((_t1uint32 <= 65535U) ? _t1uint32 : 65535U));
  /* Amf_calc: sequence call #8  */
  _t1uint16
     = ((_Amf_flgAirMassPerStrOverride) ? _Amf_mAirPerStrMan : ((_Amf_flgSwitch2AirMassFlowMeter) ?
_Amf_mAirPerStroke_AirMassFlowMeter : _Amf_mAirPerStrCalc));
  _t1uint32 = _t1uint16 * (uint32)Epm_rpmEngSpd__AIRMASSFLOW_IMPL_Amf_calc;
  /* assignment to Amf_mAirCalc: min=0, max=65535, hex=256phys+0, limit=(maxBitLength: true, assign: true),
zero incl.=true  */
  Amf_mAirCalc__AIRMASSFLOW_IMPL_Amf_calc
     = (uint16)(((_t1uint32 <= 372821333U) ? _t1uint32 * 9U / 51200U : 65535U));
  /* Amf_calc: sequence call #10  */
  _t1uint32 = (uint32)_Amf_mAirPerStroke_AirMassFlowMeter * 625U >> 3;
  _t1uint32
     = ((_Amf_facMRelAir100 == 0U) ? _t1uint32 : _t1uint32 / _Amf_facMRelAir100);
  /* assignment to Amf_facRelAir_AirMassFlowMeter: min=0, max=1280, hex=512phys+0, limit=(maxBitLength: true,
assign: true), zero incl.=true  */
  _Amf_facRelAir_AirMassFlowMeter = (uint16)(((_t1uint32 <= 1280U) ? _t1uint32 : 1280U));
  /* Amf_calc: sequence call #11  */
  /* assignment to Amf_facRelAir: min=0, max=1280, hex=512phys+0, limit=(maxBitLength: true, assign: true),
zero incl.=true  */
  Amf_facRelAir__AIRMASSFLOW_IMPL_Amf_calc
     = ((_Amf_flgRelAirOverride) ? _Amf_facRelAirMan : ((_Amf_flgSwitch2AirMassFlowMeter) ?
_Amf_facRelAir_AirMassFlowMeter : _Amf_facRelAirCalc));
  /* Amf_calc: sequence call #12  */
  /* assignment to Amf_mAirPerStr: min=0, max=65535, hex=65536phys+0, limit=(maxBitLength: true, assign: true),
zero incl.=true  */
  Amf_mAirPerStr__AIRMASSFLOW_IMPL_Amf_calc
     = ((_Amf_flgAirMassPerStrOverride) ? _Amf_mAirPerStrMan : ((_Amf_flgSwitch2AirMassFlowMeter) ?
_Amf_mAirPerStroke_AirMassFlowMeter : _Amf_mAirPerStrCalc));
  /* Amf_calc: sequence call #15  */
  _t1uint32 = (uint32)Epm_rpmEngSpd__AIRMASSFLOW_IMPL_Amf_calc * 45U >> 8;
```

```
   _t2uint32 = (uint32)Amc_AirMassFlow__AIRMASSFLOW_IMPL_Amf_calc * 2000U;
   _t2uint32 = ((_t1uint32 == 0U) ? _t2uint32 : _t2uint32 / _t1uint32);
   /* assignment to Amf_mAirPerStroke_AirMassFlowMeter: min=0, max=65535, hex=65536phys+0, limit=(maxBitLength:
 true, assign: true), zero incl.=true  */
   _Amf_mAirPerStroke_AirMassFlowMeter = (uint16)(((_t2uint32 <= 65535U) ? _t2uint32 : 65535U));
   /* send messages implicitly */
   {
      DisableAllInterrupts();
      Amf_facRelAir = Amf_facRelAir__AIRMASSFLOW_IMPL_Amf_calc;
      Amf_mAirCalc = Amf_mAirCalc__AIRMASSFLOW_IMPL_Amf_calc;
      Amf_mAirPerStr = Amf_mAirPerStr__AIRMASSFLOW_IMPL_Amf_calc;
      EnableAllInterrupts();
   }
}
/* ----------------------------------------------------------------------------
 * END: DEFINITION OF PROCESS 'AIRMASSFLOW_IMPL_Amf_calc'
 ****************************************************************************/
#endif
/* END: ASCET REGION "Process Definition 'Amf_calc'" */


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

3 Systemconstant-Parameter-Variable-Classinstance-Structure

3.1 Parameter

Name	Access	Long name	Mode	Type
Amf_facRelAirMan	READ-WRITE	Relative air charge - manually Calibrateable	local	VALUE
Amf_flgAirMassPerStrOverride	READ-WRITE	Codeword - switches between ECU calculated engine air mass and manually Calibrateable one	local	VALUE
Amf_flgRelAirOverride	READ-WRITE	Codeword - switches between ECU calculated engine air mass and manually Calibrateable one	local	VALUE
Amf_flgSwitch2AirMassFlowMeter	READ-WRITE	Codeword - switches between ECU calculated engine air mass and manually Calibrateable one	local	VALUE
Amf_mAirPerStrMan	READ-WRITE	Calibrateable engine air mass per stroke	local	VALUE
Amf_pManiEngSpd2facRelAirCorr_MAP	READ-WRITE	Engine air mass dependent on engine speed and manifold pressure	local	MAP
Amf_rpmEngSpd2facTempCorr_CUR	READ-WRITE	Correction factor of engine air temperature dependent on engine speed	local	CUR
Amf_shcSpecGasConst_C	READ-WRITE	Specific gas constant	local	VALUE
Amf_volEngDisp_C	READ-WRITE	Engine displacement	local	VALUE

3.2 Variable

Name	Access	Long name	Mode	Type
Amc_AirMassFlow	READ-ONLY	Engine Air Massflow	import	VALUE
Amf_atAirIntake	READ-ONLY	Air temperature at air intake	local	VALUE
Amf_facMRelAir100	READ-ONLY	Factor for air mass per stroke calculation	local	VALUE
Amf_facRelAir	READ-ONLY	Relative air charge	export	VALUE
Amf_facRelAirBase	READ-ONLY	Relative air charge - basis value dependent on manifold pressure	local	VALUE
Amf_facRelAirCalc	READ-ONLY	Buffer for calculating the relative air charge	local	VALUE
Amf_facRelAirTempCorr	READ-ONLY	Correction factor - ratio between standard temperature and real temperature at air intake	local	VALUE
Amf_facRelAir_AirMassFlowMeter	READ-ONLY	Exemplary description for Amf_facRelAir_AirMassFlowMeter	local	VALUE
Amf_mAirCalc	READ-ONLY	Engine air mass per stroke	local	VALUE
Amf_mAirPerStr	READ-ONLY	Exemplary description for Amf_mAirPerStr	export	VALUE
Amf_mAirPerStrCalc	READ-ONLY	Calculated engine air mass per stroke	local	VALUE
Amf_mAirPerStroke_AirMassFlowMeter	READ-ONLY	Engine air mass per stroke	local	VALUE
Amf_rohAirDen	READ-ONLY	Calculated engine air density	local	VALUE
Ctc_atCool	READ-ONLY	Engine Coolant Temperature	import	VALUE
Epm_rpmEngSpd	READ-ONLY	Engine speed	import	VALUE
Mpc_pManiFilt	READ-ONLY	Manifold air pressure - filtered - temperature compensated	import	VALUE
Mtc_atMani	READ-ONLY	Manifold air temperature	import	VALUE