BRAUMAT

Programmers-Guide for Braumat

REVISON

Ver	Date	Prepared	Checked	Approved	Description
01	16.5.2012	S. Schranner			

CONTENT

1 Purpose

This document is focussed on programmers, who want to make automation for breweries or Batch oriented processes using BRAUMAT. In this document the Main-Applications are described. For special applications please look in the manual for more Details.

2 Area

A geographical part of a factory site like:

- Brew house
- Fermentation-Cellar
- Filtration etc

A (redundant) Server for control and monitoring, recipe management, order management, reporting and optional Route-Control and maximal 32 Operator Stations (IOS), maximal 16 PCU for executing the process-control and locking programs. To realize automation and control functions the system provides each AS (PCU) with a Set of modules (fixed quantity, see System Manual Chapter 8.2). System functions can be implemented in BRAUMAT in accordance with S88.1.
In the Area you can define different **Units**

3 Units (Sequence)

A collection (up to 64 per AS) of associated control modules and/or equipment modules and other process equipment in which one or more major processing activities can be conducted. This term applies to both the physical equipment and the equipment entity.

For the Brew house it can be

- Malt-Plant
- Mill
- Mash-Tun Kettle (MTK)
- Lauter Tun (LT)
- Word-Kettle (WK)
- Whirlpool (WHP)
- CIP

For the Cellar:

- You can make for each Tank another Unit (Sequence)
- CIP Tank
- CIP Line

For the Filtration:

- Buffer-Tank
- PVPP
- Kieselguhr- Filter etc

3.1 Programming the Units in STEP 7

For each Unit you have a corresponding STEP 7-Block which is active all 1 second, independent of the production running or not.

Unit 1: FB 1001
Unit 2: FB 1002
…
Unit 64: FB 1064

Programming : DFM-Activation, Unit-Timer, Permanent Condition etc. (more Details later)
Each Unit has different Steps (GOPs, EOPs)

3.1.1 Default unit FB

Example: Default unit-FB (STL view)

```
Netzwerk 1 :  Sequence Running Condition
=============================================================================

      U     "RLO1"                   M108.1            -- Result of logic operation = 1
      =     "SEQU003_DB"             M640.2            -- SEQU003 permanent condition


Netzwerk 2 :  Action before GOP is running
=============================================================================

      U     "FBGO"                   M101.4            -- Last call of sequencer-FB (after EOP)
      SPB   AFTE


      BEA

Netzwerk 3 :  Action after GOP was running
=============================================================================

AFTE: UN    "FBGO"                   M101.4            -- Last call of sequencer-FB (after EOP)
      BEB

      BEA
```

3.1.2 Global flags and interfaces

The following flags can be used and / or modified in the unit- blocks.

Interface flags:	UNIT	1	2
	Permanent condition	*F 640.0*	*640.1*
	Sequence running:	*F 656.0*	*656.1*

Inside of the Unit-DB 725 there is a global interface used by the active unit . This structure starts at DBW100 to DBW198:

Interface flags: *"SEQ".u.STATUS.boSeqRun* *DB725.DBX 101.0*
"SEQ".u.STATUS.boSeq_Held *DB725.DBX 102.0*
"SEQ".u.STATUS.boMan *DB725.DBX 103.2*

3.1.3 Real unit FB as a example

The following rules or options for the designing of a unit-FB are defined:

- In the first network you can define the so called permanent condition (or running condition) of an units
- Then you enable the (monitoring)-time of the unit
- Then you can call the DFM's which are used in this Unit
- Then you can start and stop of the trending-group
- Then you can handle the hold-button (a button to switch off the outputs during Automatic).

Example: Real unit-FB (SCL code)

```
FUNCTION_BLOCK FB 1001
TITLE = Sequence User-FB Interface (1 sec)
//History:
//Date      Name            Action
//-------------------------------------------------------------------
//06.11.12  Author
// ----------------------------------------------------------------------
// ATTENTION:  No VAR_IN, VAR_OUT, VAR_IN_OUT, VAR-Variables allowed here.
AUTHOR : SCHRANNER
FAMILY : SISTARS7
NAME : TA01_FB
VERSION : 5.2

VAR_TEMP
  byTempVar : BYTE ;
END_VAR
BEGIN

NETWORK
TITLE = Sequence Running Condition
// here you can define Interlocks of Sequences, for example CIP and Production
        AN      F    657.1;            // Sequence 10 running;
         =      F    640.0;            // SEQU001 permanent condition

// set to A+ with Start of Sequence
        AN      DB725.DBX  101.0;    // "SEQ".u.STATUS.boSeqRun
        FP      M.xy ;               // use any flag as Impulse
        S       DB725.DBX  102.0;    // "SEQ".u.STATUS.boSeq_Held
        R       DB725.DBX  103.2;    // "SEQ".u.STATUS.boMan
```

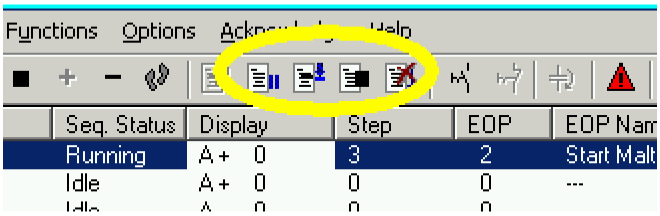
```
NETWORK
TITLE = enable time of sequence
// here you enable the (monitoring)-time of the sequence.
// The Flag is defined in the Parameterization
      A     F  656.0;          // Sequence 1 running
      AN    F  712.0;          // and Not Manual, manual group 1
      =     F   2.0;           // enable monitoring-time Sequ 1

NETWORK
TITLE = Action before GOP is running
      A     F  101.4;          //Last call of sequencer-FB (after EOP)
      JC    AFTE;

NETWORK
TITLE = CALL DFM
      CALL  FC  736 (          // call of DFM which are used in this Sequence
      iDfm           := 1);
      CALL  FC  737 (          //activate DFM
      iDfm           := 1);

NETWORK
TITLE =start and stop of trending-group
//Curve-group corresponds to the group in the Measurement-Description-List
      CALL  FC  695 (          // start trend
            boRUN         := F  102.1          ,// sequence is running
            boACTSEQ      := TRUE,
            iID           := 1                 ,// Curve-group
            iSEQID        := 0,
            byRECCAT      := B#16#0,
            byYEAR        := B#16#0,
            iORDER_NO     := 0,
            iBATCH_NO     := 0,
            iRECIP_NO     := 0);

NETWORK
TITLE = end Action before GOP was running
      BEU;                              // do not delete!!!

NETWORK
TITLE = Action after GOP was running
AFTE: AN    F  101.4;
      BEC;

NETWORK
TITLE = Reset all ICM in case of pressing Hold
      A     DB725.DBX 110.1;        // UN    "SEQ".u.CTRL.boHeld
      R     F  128.0;       // Reset all ICM used in this Unit for Hold-operation
      R     F  128.1;       // Reset ICM 2

END_FUNCTION_BLOCK
```

Outputs could be also reset with
A DB725.DBX 110.5; // "SEQ".u.CTRL.boPaused
as you like

3.1.4 Activate a S88-status

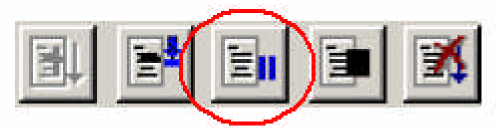
To deactivate all outputs (see last network in the example-FB) of the units in case of emergency, you can do the following:

- Activate the S88-commands in the SEQUENCE-control-application



Change in c:\windcs\sys\seqctrl.ini:
[MainWin]
EOPAdmin=255

- Push the HOLD - or PAUSE - button



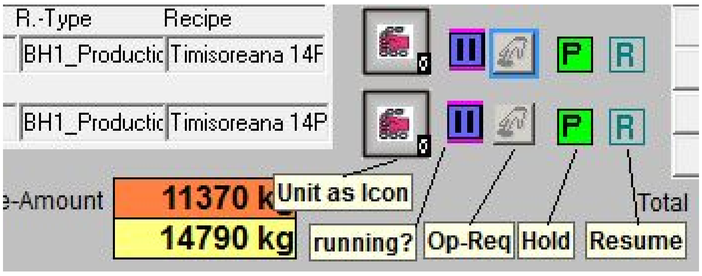
3.1.5 For to Reset Outputs with hold from the S88-Buttons

This button in the Unit-Faceplate or Sequence-overview

Program in the FB100x (x according to Unit) at the network “Action after GOP was running”

```
A    DB725.DBX  110.1    //   "SEQ".u.CTRL.boHeld
 R    "ICM1.001_CA" and other ICM which are involved in the Unit
```
Hold should also reset the monitoring and Step-Time

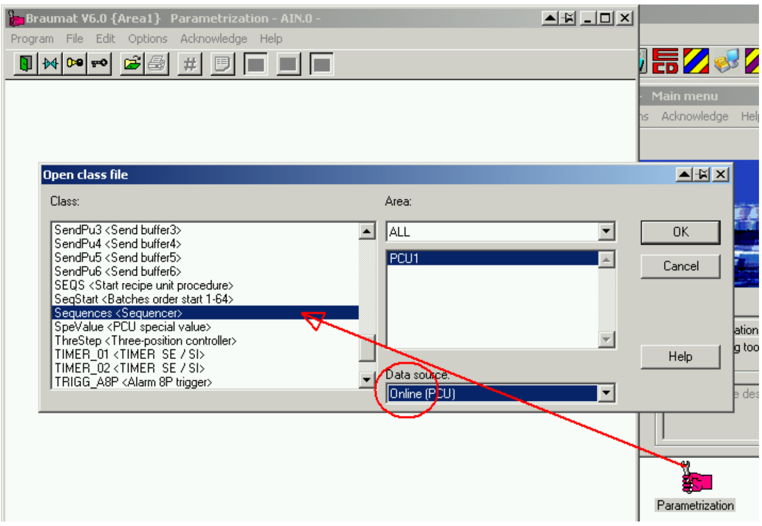
3.1.6 Hold and Release in the process-Image



For to display flashing Hold in the screen (may be next to Sequence-Faceplate) select Flag R_Held of the Unit-Faceplate
For to release from Hold show with another Button R_Restart-Flag of the Unit. May be with Operator-Level “-1” to avoid a window popping up.
For to show whether there is a Stop of the Sequence show flag R_Running of the Unit flashing
To advise the Operator about necessary action, you can make another symbol with flashing Operator-Request linked to Flag OpReq of the Unit.
A separate Button for to show the Icon of the Unit-Faceplate is very useful
In the Software outside the Unit check for AN "SEQ".au[x].CTRL.boHeld for the Held-Signal.

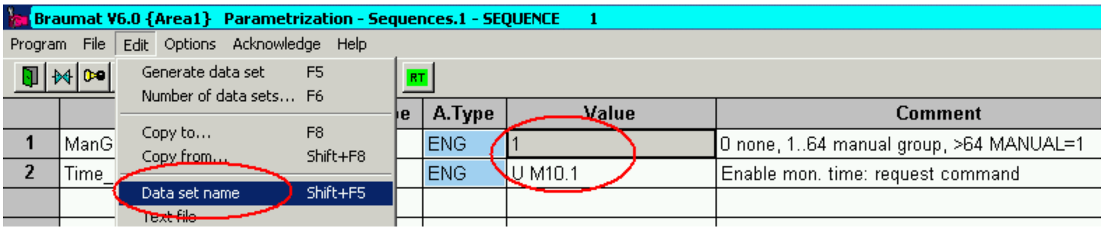
3.2 Parameterization of Units

You have to setup some parameters for the Unit in the **Parameterization.**
Here you open the dialog for the according data-set.



Sequence 1 corresponds to Unit 1 etc. The data-set-name is the name of the Unit (e.g. Malt..).

Here you have to input the manual-group (you can make the same like the Unit-number) and a free enable-Flag (e.g. F 10.1 in this case) for the Monitoring-Time.


Braumat V6.0 {Area1} Parametrization - Sequences.1 - SEQUENCE 1
Program File Edit Options Acknowledge Help
Generate data set F5
Number of data sets... F6
Copy to... F8
Copy from... Shift+F8
Data set name Shift+F5
Text file
A.Type
Value
Comment
1 ManG
ENG
1
0 none, 1..64 manual group, >64 MANUAL=1
2 Time_
ENG
U M10.1
Enable mon. time: request command

3.3 EXCEL-Tool for Units

In *\windcs\EXCEL\Param_PCU001_1.xls* you find templates for the parameterization of all objects like SEQU for sequencers and so on. The parameters can exported from EXCEL into the BRAUMAT parameter data blocks.

	A	B	C	D	E	F	G	H	I	J	K
1	**PCU**	**1**									
2	**Class_RecV3**	**SEQU**									
3	**Class_RecV5**	**Sequences**									
4	**Amount**	**64**									
5	**Max**	**64**									
6	**Export**	**E**									
7											
8	Def_RecV3	No	NAME	Sper	ESGS	MESS	HZuo	RTyp	WTueS	QBI	Addr
9	Def_RecV5	No	NAME	Disable	_	_	ManGroup	BA_RecTyp	Delay_SP	Time_Rel	Addr
10											
11	**Filter**	**SEQU-Nr**	**SEQU-Name**	**Edit disable**	**Group error ESG**	**Group error MEAS**	**Manual Group**	**Recipe Type**	**Wait time target value (sec.)**	**Release Mon-Time: Step-7 query command**	**Marker Permanent condition and Start TA**
12	E	1	Malt intake	0	0	0	0	1	10	U M108.1	Sequences 1
13	E	2	Malt outtake	0	0	0	0	2	10	U M108.1	Sequences 2
14	E	3	Sel Mash tun	0	0	0	0	2	10	U M108.1	Sequences 3
15	E	4	Millstar 1	0	0	0	0	2	10	U M108.1	Sequences 4
16	E	5	Millstar 2	0	0	0	0	2	10	U M108.1	Sequences 5
17	E	6	Mash tun 1	0	0	0	0	2	10	U M108.1	Sequences 6
18	E	7	Mash tun 2	0	0	0	0	2	10	U M108.1	Sequences 7
19	E	8	CaCl -> MBF	0	0	0	0	2	10	U M108.1	Sequences 8
20	E	9	Lautertun	0	0	0	0	2	10	U M108.1	Sequences 9
21	E	10	Lauter Parameter	0	0	0	0	2	10	U M108.1	Sequences 10
22	E	11	Wort prerun tank	0	0	0	0	2	10	U M108.1	Sequences 11
23	E	12	Wort kettle	0	0	0	0	2	10	U M108.1	Sequences 12
24	E	13	Hop Dosing	0	0	0	0	2	10	U M108.1	Sequences 13
25	E	14	Whirlpool	0	0	0	0	2	10	U M108.1	Sequences 14
26	E	15	Trubtank	0	0	0	0	2	10	U M108.1	Sequences 15
27	E	16	CIP Brewhouse	0	0	0	0	3	10	U M108.1	Sequences 16
28	E	17	CIP MakeupDosing	0	0		0	4	10	U M108.1	Sequences 17

4 Steps (EOP: Equipment Operations- / TOP: Technical operations- / GOP)

Equipment operations (Steps) are self-contained functions used in the units.
Max. 999 EOPs per PLC are possible.

For the Mash-Tun it can be:

- Start-up
- Heating
- Rest
- …

4.1 Programming the EOPs in STEP 7

For each Step you have a corresponding Step 7-Block which is active all second.

Step 1 (EOP 1): *FC 1001,*
Step 2 (EOP 2): *FC 1002,*
…
Step 999 (EOP 999): FC 1999,

Programming: Start Step-Time, activate ICM (Motor, Valve), check for next-Step-Condition (Transition). More Details later!

4.1.1 Default EOP FCB

A default EOP FC is not existing in the system. When a recipe will not found a responsible EOP FC in the PLC, the reaction is the same like using a NOP. So nothing will called and nothing will happened. The recipe jumps over this step into the next one.

4.1.2 Global flags and interfaces

You can use the same flags and interfaces , which was described at the unit-FB (3.1.2.)

4.1.3 Structure of an EOP FCs

The following rules or options for the designing of a unit-FB are defined:

- There is no special call of the EOP-FC necessary. The system will make the call, when the EOP is used in the recipe.
- In the first network you can take over the accu values for the current STEP- and EOP – number.
- Also the last RLO can used for switching on a valve.
- The next networks can used for any individual programming of activating ICMs
- The structure off the last network is not free. We need a result for the “next step condition”.

4.1.3.1 First network

Example: activating valves and motors

```
=    F  128.0;    // Activate ICM 1.1
=    F  128.1;    // Activate ICM 1.2
```

4.1.3.2 Any network before the last one

Example: Set ICM output depending on the DFM –comparison result

```
// When counter setpoint reached, stop motor
AN   F 102.3      // EOP start impulse
AN   F 728.0      // DFM 0.1 Result-Flag
=    F 129.0      / /ICM 1.1 activate Output
```

Example: Enable timer, when reaching the setpoint

```
AN   F 102.3      // EOP start impulse
AN   F 760.0      // DFM1.1 Result-Flag Timer
=    F 10.0       // The enable-flag for the DFM-Timer, defined in the
                  // Parameterization.
```

4.1.3.3 Last network

The last network in the block is reserved for the "**Next-Step-Condition"**.

Example: Reaching setpoint of a DFM

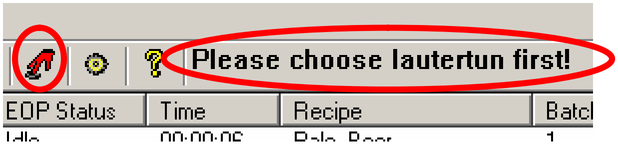
```
A    F 761.1      // Result flag from setpoint DFM1.10
A    F 729.1      // Result flag from counter DFM0.10
```

If the result-flag at the end is "1", then BRAUMAT is calling the FC-Block once again for one cycle to reset all outputs (by the command "*AN F 102.3*").
After this the system make the call of the next EOP-FC immediately with the flag *F 102.2 = 1*.
If necessary you can reset all ICMs in step1 and activate them again in the next step

4.1.4 Special use cases inside the EOP FCs

4.1.4.1 Handling of “operator requests”

The flag F 1001.6 (“BEDA”) is used to display a operator request message inside the sequencer control application. If the flag 101.6 is set in an EOP - block (FC1001…), the operator will see this message and the following activated button:



The text can be modified in the recipe – editor at EOP-definition.
The operator have to acknowledge the message by pushing the button

Example: Activate the OpReq. at the beginning to the step

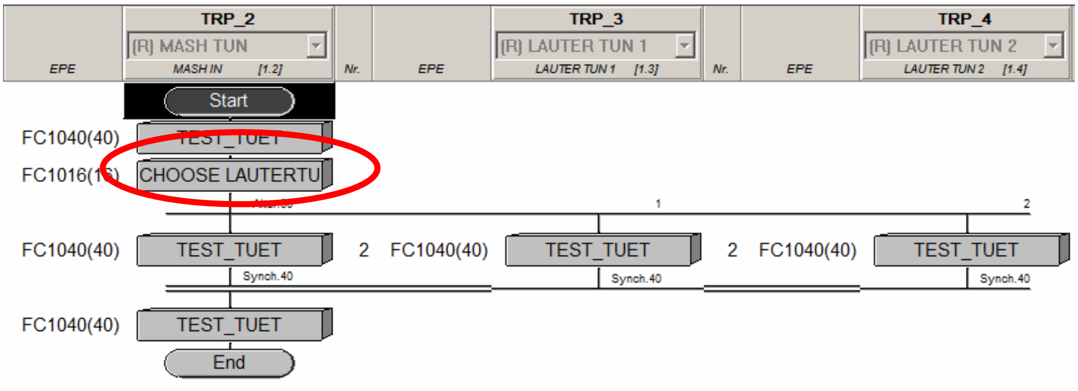
```
A    F 102.2    // EOP start impulse
S    F 101.6    // Operator request
```

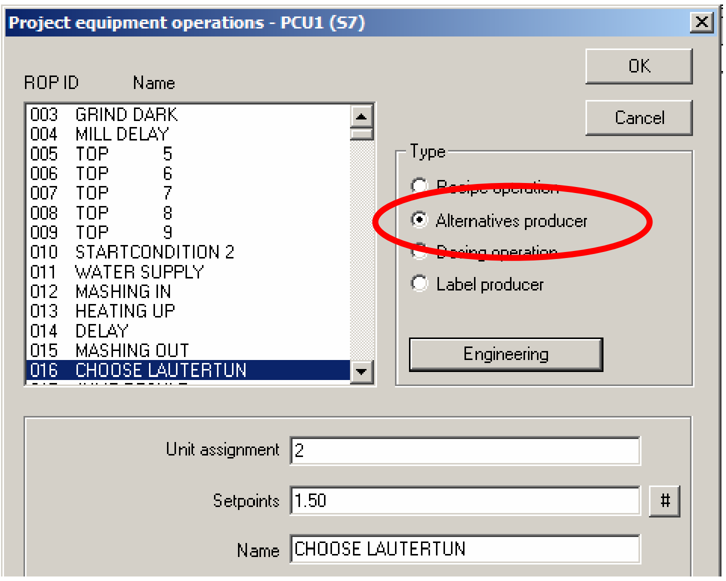
As a “next step condition” command in the last network you can use the following:

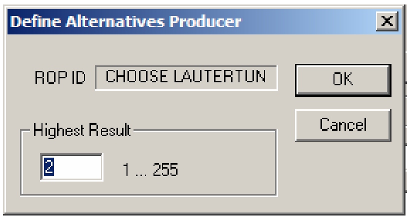
```
AN   F 101.6    // Operator request acknowledged
```

4.1.4.2 Handling of “alternatives”

An alternative inside a recipe procedure will defined in the recipe editor:







- You declare one EOP as alternative - producer in the EOP-properties with a highest result of alternatives.
- You insert a alternative - line with the connection to different units
- You insert the alternative - numbers at the alternative exits
- You have to reconnect all units with an OR – synchronization

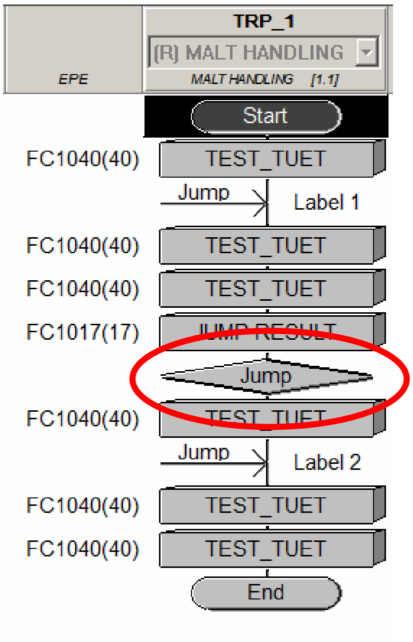
In the EOP – blocks (FC1001…) you only need to transfer the number of the alternative a special reserved data byte from the sequencer dataset.

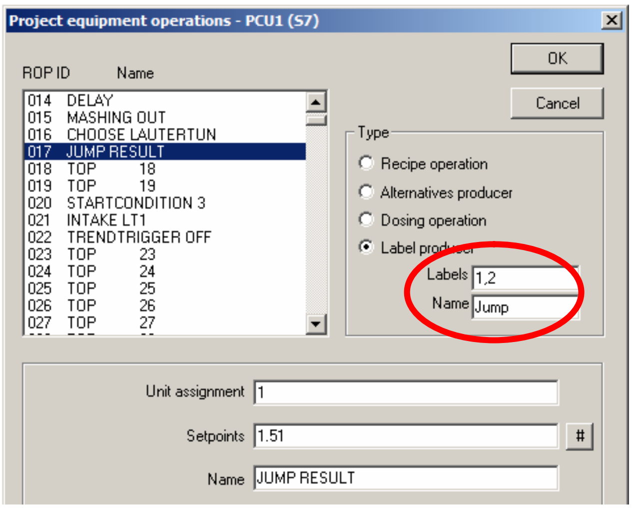
Example: Activate one unit depending on a DFM setpoint

L DB737.DBD1380 // Alter- number “DFM1”.au[50].rSPVal
T DB725.DBB 177 // Alter – result ”SEQ”.u.byAlterResult

4.1.4.3 Handling of “jumps”

A step jump inside a recipe unit procedure will defined in the recipe editor:





- You declare one EOP as label-producer in the EOP-properties with different Label (Jumps)
- You insert a jump after this EOP
- You insert one or more labels at the EOP’s you want to jump to
- You can not jump over a synchronisation, at the very beginning or end.

In the EOP – blocks (FC1001…) you only need to transfer the number of the label into a special reserved data byte from the sequencer dataset.

Example: Jump to Label 1, when a special condition is active

```
        A      M  10.0    // condition for the Jump
        JCN    A001
        L      1              // Label-Number
        T      DB725.DBB 177  // Jump – result ”SEQ”.u.byAlterResult
A001:   NOP 0
```

4.1.5 Real EOP-FC as example

Example: EOP-FC from the training (STL code)

```
FC1002 :  TOP2 - Grind pale
Comment:

Network 1: Start motor from mill
Comment:

      =     "Milling_CA"               M128.2            -- ICM1.003 automatic activation command


Network 2 : open outlet valve
Comment:

      AN    "GSTO"                     M102.3            -- EOP stop impulse
      AN    "PaleMalt_DFM"             M728.0            -- DFM0.001: digital function module  count
      =     "Silo1_CA"                 M128.0            -- ICM1.001 automatic activation command

Network 3 : Next step condition
Comment:

      A     "PaleMalt_DFM"             M728.0            -- DFM0.001: digital function module  count
```

Example: Real EOP - FB (SCL code)

FUNCTION FC 1001 : VOID
TITLE = EOP-Name
//History:
//Date   Name   Aktion
//---
//20.07.12 Your Name   V1.0
AUTHOR : SCHRANNER
FAMILY : SISTARS7
NAME : choose
VERSION : 5.3

BEGIN

NETWORK
TITLE =Activate Output unconditionally
// here all Output are activated and reset at the end of the Step
=   F 128.0;   // add all command- automatic-flags like this in here. ICM 1.1
=   F 137.1;   // ICM 1.74 Command Automatic
=   F 227.5;   // ICM 4.30 Command Automatic

```
NETWORK
TITLE =Activate output conditionally
// here an output will activated conditionally and reset at the end of the step with flag F 102.3
      AN     F  102.3;      // EOP start impulse
      AN     F  728.0;      // DFM 0.1 Result-Flag, counter finished
      =      F  128.1;      // ICM 1.1 activate output

NETWORK
TITLE =Enable step-time
// here the step-time is enabled, as long as it is not elapsed.
      AN     F  102.3;      // EOP start impulse
      AN     F  760.0;       // DFM1.1 Result-Flag Timer
      =      F  10.0;       // The enable-flag for the DFM-Timer, defined in the Para.

NETWORK
TITLE =Switch Output on delayed
// Use the Timer-Values in DB709 to setup the Timer. They can be modified direct
// in BRAUMAT with the Function "Special values".
      AN     F 102.3;             // EOP stop impulse
      L      DB709.DBW 2;         // Timer-value from Special Values 1; 2005 = 5 sec
      SD     T 57;                // Start Timer
      AN     F  102.3;            // EOP stop impulse
      A      T 57;                // Delay Start
      =      F  129.3;            // Output ICM x

NETWORK
TITLE =Operator-Request
// there will be a symbol for operator-request popping up in the screen. It must be
// confirmed by the operator. The text can be modified at the recipe, EOP-definition.
      A      F  102.2;      //EOP start impulse
      S      F  101.6;      //Operator acknowledge request

NETWORK
TITLE =Generate a Alarm
// if the output of the DFM is set, a alarm is generated
      AN     F  102.3;      // EOP stop impulse
      A      F  760.1;      // DFM1.2 Result-Flag Temperature
      =      F  888.0;      // Alarm-Flag

NETWORK
TITLE = Next Step Condition
// Result must be 1 for going to the next Step;
      AN     F  102.2;      // this is necessary if DFM is the next step condition
      A      F  728.0;      // DFM 0.1 Result-Flag
      A      F  760.0;      // Time is finished
      AN     F 101.6;        // Operator has acknowledged

END_FUNCTION
```

4.2 Parameterization of EOPs

There is no special parameterization for the EOPs necessary in the PLCs. Later, when talking about recipes, we have to put in some parameters.

5 Setpoints (DFM: Digital Function Modules)

Each step has different setpoints defined in the **Digital-Function-Modules (DFM)** of BRAUMAT.
Digital-function-modules can be used for a variety of functions for processing the set values. They have an interface by a flag to trigger for example the next-step-conditions etc.
In each step you can have up to 20 DFMs.
In one AS (PCU) max.1024 setpoints for the control of the process can be used.

DFMs are used as:

- Timers
- Analogue Values with comparison
- Selector (Decoder)
- Counters

5.1 DFM as Counter (Count Pulses)

5.1.1 Programming a counter in STEP7

Call FC736 in the Unit- FB100x to activate the DFM.
Use the FB 1222 – FB 1224 for programming the counting-pulses. The Pulse-length must be at least 100 msec!

Example: Count the pulses in the DFM, if the pump is running

```
A    I 64.0     //motor is running ICM1.1
A    I 10.0     //Counting-pulse Input
=    F 984.0    //DFM 0.1 counting pulse
```

The result-flags of the counters start at F *728.0*

5.1.2 Parameterization of counters

You go to the Parameterization and select DFM0 (counter).
You setup the dataset-name and the rest can be like that:

Braumat V6.0 {Area1} Parametrization - DFM0.1 - DFM0 1

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	SOLL	I16	ENG	0	Target value low word
2	SOLL_DINT	I32	ENG	0	Target value double integer
3	InitActVal	B1	ENG	0	SP_CM actual value init. if target value from recipe=#
4	Grenze_DINT	I32	ENG	0	Switching limit double integer
5	Richtung	B1	ENG	0	0=up counter, 1=down counter
6	Summation	B1	ENG	0	0=non accumulating, 1=accumulating
7	Art	B1	ENG	0	0=PSK is increment, 1=PSK is reduction
8	PSK	I16	ENG	1	Increment / reduction

see “BRAUMAT Manual Blocks S7”, “DFM” for more information

5.2 DFM as Timer

5.2.1 Programming a timer in STEP7

Call FC737/738 in the FB100x to activate the DFM.
Use the Step-Blocks FC1001… for programming the timer.

Example: Start a waiting time for the next step transition

```
// here the Step-Time is enabled, as long as it is not elapsed.
AN   F 102.3     // EOP start impulse
AN   F 760.0     // DFM1.1 Result-Flag Timer
=    F 20.0      // enable-flag for the DFM-Timer, defined in the param.
```

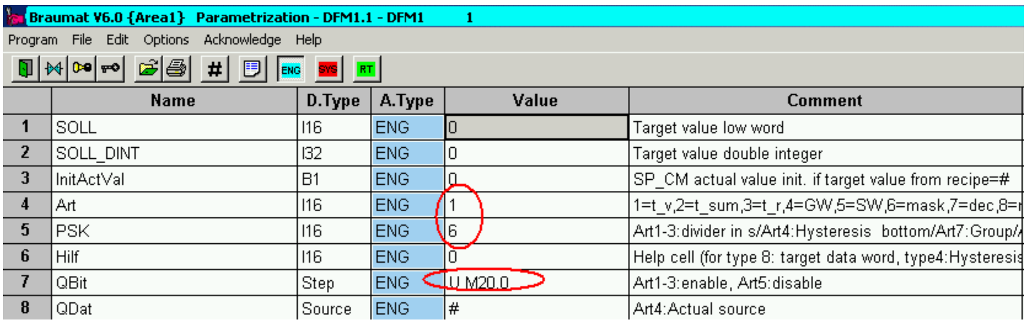
The result-flags of the timers starts at *F 760.0.*

In the transition program use the following command:

```
A    F 760.0          // DFM 1.1 Timer elapsed
```

5.2.2 Parameterization of timers

You go to the Parameterization and select DFM1-3.
You setup the dataset-name and the rest can be like that:


Braumat V6.0 {Area1} Parametrization - DFM1.1 - DFM1 1

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	SOLL	I16	ENG	0	Target value low word
2	SOLL_DINT	I32	ENG	0	Target value double integer
3	InitActVal	B1	ENG	0	SP_CM actual value init. if target value from recipe=#
4	Art	I16	ENG	1	1=t_v,2=t_sum,3=t_r,4=GW,5=SW,6=mask,7=dec,8=r
5	PSK	I16	ENG	6	Art1-3:divider in s/Art4:Hysteresis bottom/Art7:Group/
6	Hilf	I16	ENG	0	Help cell (for type 8: target data word, type4:Hysteresis
7	QBit	Step	ENG	U M20.0	Art1-3:enable, Art5:disable
8	QDat	Source	ENG	#	Art4:Actual source

Art = 1 corresponds to type = Time

PSK = 6 corresponds to counting up all 6 seconds, makes a time-grid like 1.5 min = 1 minute and 30 seconds.

QBit = *A F 20.0* is a free enable-flag for a waiting-time (e.g. waittime to the next Step).

See “BRAUMAT Manual Blocks S7”, “DFM” for more information

5.3 DFM as a Analogue –Setpoint

5.3.1 Programming an Analogue - Setpoint

Call FC737/738 in the FB100x to activate the DFM.
Use the Step-Blocks FC1001… for programming the Analogue – Setpoint.

Example: Check a Temperature for switch on / off of an output

AN	*F 102.3*	*// EOP start impulse*
AN	*F 760.1*	*// DFM1.2 Result-Flag Temperature*
=	*F 128.0*	*// ICM1.1 Output*

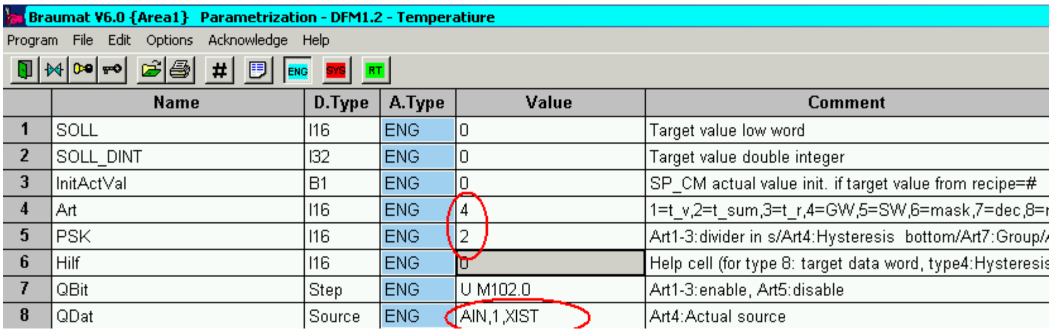
The result-flags of the DFM 1.2 is F *760.1*

In the transition program use the following command:

A	*F 760.1*	*// DFM 1.2 Temperature is ok*

5.3.2 Parameterization of analogue setpoints

You go to the Parameterization and select DFM1-3.
You setup the dataset-name and the rest can be like that:



	Name	D.Type	A.Type	Value	Comment
1	SOLL	I16	ENG	0	Target value low word
2	SOLL_DINT	I32	ENG	0	Target value double integer
3	InitActVal	B1	ENG	0	SP_CM actual value init. if target value from recipe=#
4	Art	I16	ENG	4	1=t_v,2=t_sum,3=t_r,4=GW,5=SW,6=mask,7=dec,8=r
5	PSK	I16	ENG	2	Art1-3:divider in s/Art4:Hysteresis bottom/Art7:Group/
6	Hilf	I16	ENG	0	Help cell (for type 8: target data word, type4:Hysteresis
7	QBit	Step	ENG	U M102.0	Art1-3:enable, Art5:disable
8	QDat	Source	ENG	AIN,1,XIST	Art4:Actual source

Art =	4 corresponds to type = analogue setpoint
PSK =	2 corresponds to the hysteresis
QDat =	“AIN,1,XIST” corresponds to the source of the analogue input.

See “BRAUMAT Manual Blocks S7”, “DFM” for more information.

5.4 DFM as a Decoder

5.4.1 Programming a Decoder

With the Decoder, you can make different selections in the recipe by a text-file.

- You have 3 decoder-groups (with 64 switches each)
- Each correspond to different output-flags
- Decoder 0 = Output-flag M688.0….
- Decoder 1 = Output-flag M696.0…
- Decoder 2 = Output-flag M704.0…
- Select the decoder-Group in Parameter PSK (0-2)

Activate the DFM in the FB100x by call FC737 and DFM-number.

Example: Program in the step-blocks like this:

```
AN    F 102.3    // EOP start impulse
A     F 688.0    // Decoder 0.1 is active
=     F 129.0    // ICM 1.1 activate Output

AN    F 102.3    // EOP start impulse
A     F 688.1    // Decoder 0.2 is active
=     F 129.1    // ICM 1.2 activate Output
```

5.4.2 Parameterization of a decoder

You go to the Parameterization and select DFM1-3.
You setup the dataset-name and the rest can be like that:

Braumat V6.0 {Area1} Parametrization - DFM1.3 - Decoder Mode

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	SOLL	I16	ENG	0	Target value low word
2	SOLL_DINT	I32	ENG	0	Target value double integer
3	InitActVal	B1	ENG	0	SP_CM actual value init. if target value from recipe=#
4	Art	I16	ENG	7	1=t_v,2=t_sum,3=t_r,4=GW,5=SW,6=mask,7=dec,8=r
5	PSK	I16	ENG	0	Art1-3:divider in s/Art4:Hysteresis bottom/Art7:Group/
6	Hilf	I16	ENG	0	Help cell (for type 8: target data word, type4:Hysteresis
7	QBit	Step	ENG	U M102.0	Art1-3:enable, Art5:disable
8	QDat	Source	ENG	#	Art4:Actual source

Art = 7 corresponds to type = decoder
PSK = 0 corresponds to the decoder group 0
Interface-Flag for decoder 0.1: F *688.0*

See “BRAUMAT Manual Blocks S7”, “DFM” for more information.

5.5 Programming for to activate the DFM

For each DFM you use you have to make a call in the corresponding

- Unit-FB1001, FB 1002… or
- Step-FC1001, FC1002 …

```
CALL        FC 736        // activate DFM 0.1
  iDfm:= 1
CALL        FC 737        // activate DFM 1.1
  iDfm:= 1
CALL        FC 738        // activate DFM 2.1
  iDfm:= 1
```

6 ICMs (Valves, Motor, Pumps,…)

Valves and motors are handled by the **ICM**-object **(ICM = Individual control Module)** of BRAUMAT.
Max. 1024 ICMs per PCU are possible.
All ICM-modules have a fix input – and output - address, but it can be modified if needed (more Information in this document)!

6.1 Global flags and interfaces of ICMs

The flags, input- and output – addresses can be modified and called in the user-, unit- or EOP – blocks.

Interface- Flags:		ICM 1.1	1.2
Feedback on:	*I*	*64.0*	*64.1*
Feedback off:	*I*	*128.0*	*128.1*
Output:	*Q*	*64.0*	*64.1*
Command Automatic:	*F*	*128.0*	*128.1*
Interlock:	*F*	*256.0*	256.1

6.2 Parameterization of an ICM

You go to the Parameterization and select ICM1-4.
You setup the dataset-name and the rest can be like that:

Braumat V6.0 {Area1} Parametrization - ICM1.1 - V101
Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	TEILANL	I8	ENG	1	Assigned unit
2	HZUO	I8	ENG	1	0 none, 1..64 manual group, >64 MANUAL=1
3	TYP	I8	ENG	49	Type 8..13,16..21,32..38,48..53,128=locked
4	TUE	I8	ENG	10	Monitor time in sec
5	SollEvz	I16	ENG	0	Target value for turn on delay
6	SollAvz	I16	ENG	0	Target value for turn off delay
7	InvertQL	B1	ENG	0	Inverted load output
8	NoRetSig	B1	ENG	0	ICM without reply
9	WarnON	B1	ENG	0	Startup warning
10	ForceEna	B1	ENG	0	Forcing enable
11	SIM	B1	ENG	0	Simulation RE/RA
12	FltTime	HEXA8	ENG	0	Error time in sec (Bit0-3:setpoint, Bit4-7:actual value)

Teilanl =	1 corresponds to the Unit 1 or Sequence 1 (for Alarms)
Hzuo =	1 corresponds to the manual group 1 in most cases same number line Unit 1
Type =	49 corresponds to a valve with a feedback on (remains switched on in case of Alarm)
Type =	53 corresponds to a motor with a feedback on (gets switched off in case of Alarm).
TUE =	10 monitoring time in sec.

Implementation of a valve with 2 Outputs

You can make it like that:

You have 2 ICM for that. The ICM for closed is hidden (not visible in the picture) and must be always in automatic (according manual group).

Then you program:

AN "ESG1.1-QL" //Output valve open

= "ESG1.2-CA" //Automatic-Flag Valve closed

You have to transfer both Feedback open and closed the Valve for open, to get the Alarm.

6.3 Programming a different address assignment

Sometimes it is necessary to deviate from the standard - ICM - addressing. You have to do the following changes in the prepared interface - blocks

6.3.1 Different inputs

Example: Different input for ICM 1.8

```
Network 4 :  ICM1 Input
 ----------------------------------------------------------------------

      CALL  "ESG_IN_OUT_FC"            FC726              -- E/A-Rangier
       boIN  :=TRUE
       iESGGr:=1
//boESG1  :=      TRUE

Network 5:  User-Interface after ICM1-Input
 ----------------------------------------------------------------------

      A      "1856HV37RE"              I10.0              -- Absperrkl.
      =      "ESG_RE".aboESG1[8]       DB603.DBX10.7
      BEU
```

- Map at the marked point the input to the corresponding data in DB603.
- Than the original reserved ICM input can then be used elsewhere.
- For other ICM groups use the corresponding network 9, 13, 17
- Do not remove the BEU!

6.3.2 Different outputs

Example: Different output for ICM 1.8

```
Network 4 :  ICM1 Output
 --------------------------------------------------------------------------

      CALL  "ESG_IN_OUT_FC"            FC726            -- E/A-Rangierung ESG
       boIN   :=FALSE
       iESGGr:=1
//boESG1  :=      TRUE

Network 5 :  User-Interface after ICM1-Output
 --------------------------------------------------------------------------

      A     "ESG_QL".aboESG1[8]        DB605.DBX10.7
      =     Q     10.0
      BEU
```

- Map at the marked point on the output to the corresponding data in DB605.
- The original output can then NOT be used otherwise. When this is necessary, you have to make changes in the FC726.
- For other ICM groups go to corresponding network 9, 13, 17
- Do not remove the BEU!

7 ICM - Interlocks

The ICMs - interlocks are used to lock the ICM – output – signals for security reason and machine protection.

Example: No pump will open on if the corresponding valves are not opened.
Stop filling a tank when the full sensor is active.

7.1 Programming the ICM - interlock in STEP7

This logic should be programmed in the prepared interlock-FB-blocks.

ICM1_BV1_128	*FB 1226*
ICM1_BV129_255	*FB 1227*
ICM2_BV1_128	*FB 1228*
ICM2_BV129_255	*FB 1229*
:	
ICM4_BV129_255	*FB 1233*

The interlock – flags starts at *F 256.0*

Example: Interlocking the agitator

Network 1:	*1.1*		
	A	*I 66.0*	*//Aspiration is running*
	=	*F 256.0*	*//Enable ICM 1.1 Agitator*

The following rules are defined:

- The ICM - output is enabled, when the result-flag = 1
- Keep the networks in the FB 1226-1233 as predefined for display in the process-image-faceplate (Network 1 is reserved for ICM1 , etc.)

8 AIN (Analogue Input Values)

Analogue inputs are handled by the **AIN**-object of BRAUMAT.
Max. 255 AINs per PCU are possible.
All AIN-Modules have a fix input-address, but it can be modified if needed!

Example:	*AIN 1:*	*PIW 512*
	AIN 2:	*PIW 514*

8.1 Programming an analogue switch with AIN

You can use the AIN limit signals for switching on and off the e.g. inlet valve of a tank. The combination of limit and hysteresis will set the reserved flags depending on the input value.

8.1.1 Global flags and interfaces

The following flags can be used in the EOP- blocks.

Interface-Flags:	AIN	1	2
	lower limit	*F 1144.0*	*1144.1*
	upper limit	*F 1176.0*	*1176.1*

8.1.2 Examples

Example: Switch off valve in Case of max - limit

```
AN    F M1176.0    // AIN Limit  flag (full)
=     F 256.0      // Output ICM x Filling Interlock
```

8.2 Parameterization of an analogue input value

You go to the Parameterization and select AIN.
You setup the dataset-name and the rest can be like that:

Braumat V6.0 {Area1} Parametrization - AIN.1 - Temperature MTK

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	DigX_ANF	I16	ENG	0	digital start value
2	DigX_END	I16	ENG	27648	digital end value
3	XANF	I16	ENG	0	Start value
4	XEND	I16	ENG	1200	End value
5	UNTGR	I16	ENG	100	Lower limit
6	HYST_U	I16	ENG	10	Hysteresis lower limit
7	OBERGR	I16	ENG	980	Upper limit
8	HYST_O	I16	ENG	10	Hysteresis upper limit
9	TEILANL	Byte	ENG	0	Assigned unit
10	DigValType	Byte	ENG	0	AI format: 0=S7,5=S5 two's compl.,6=S5 sign+abs
11	STWE	B1	ENG	0	XIST with error: 0/1 = XANF/XEND
12	SIMU	B1	ENG	0	Simulation: 0,1 = Off/On
13	FEHL_UG	B1	ENG	1	Error lower limit
14	FEHL_OG	B1	ENG	1	Error upper limit
15	FREI_FUG	B1	ENG	0	Enable error output lower limit
16	FREI_FOG	B1	ENG	0	Enable error output upper limit
17	MLDG_SPERR	B1	ENG	0	no error message at underflow/overflow
18	LIMIT	B1	ENG	0	Limit XANF<=XIST<=XEND
19	S5Live	B1	ENG	0	S5-Typ:Drahtbrucherkennung < 2,9 mA

XANF =	0 corresponds to the start of the range
XEND =	1200 corresponds to the end of the range (1200 = 120.0 °C)
UNTGR =	100 corresponds to a low-limit for alarm etc. (100 = 10.0 %)
Hyst_U =	10 corresponds to a hysteresis of 1.0 for the low-limit
OBERGR =	980 corresponds to a high-himit for alarm etc. (980 = 98.0 %)
Hyst_O =	10 corresponds to a hysteresis of 1.0 for the high-limit

See the BRAUMAT-Manual Blocks S7, AIN for more Information

9 MVC (Measured value control)

When the input value is not coming directly from an analogue card, you can use the MVC – object. You can link the input value via parameterization, as well define the limits and hysteresis.

9.1 Programming a MVC

You can use the MVC limit signals for switching on and off the e.g. inlet valve of a tank. The combination of limit and hysteresis will set the reserved flags depending on the input value.

9.1.1 Global flags and interfaces

The following flags can be used in the EOP- blocks.

Interface-Flags:	**MVC**	1	2
	limit 1	*F 856.0*	*856.1*
	limit 2	*F 872.0*	*872.1*

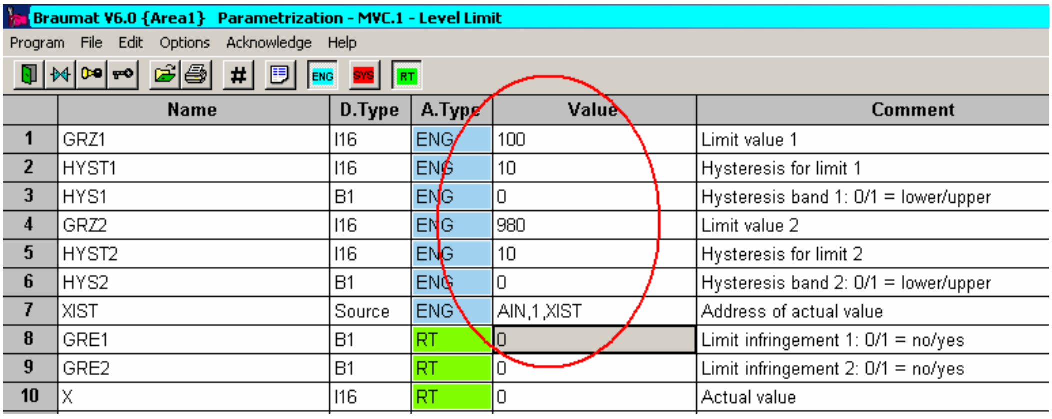
9.1.2 Examples

Example: Switch off valve in Case of max – limit

```
AN   F 856.0   // MVC 1 Limit 1 (empty)
S    F 128.0   // Output ICM x Filling
A    F 872.0   // MVC 1 Limit 2 (full)
R    F 128.0   // Output ICM x Filling
```

9.2 Parameterization of an analogue input value

You go to the Parameterization and select MVC.
You setup the dataset-name and the rest can be like that:



	Name	D.Type	A.Type	Value	Comment
1	GRZ1	I16	ENG	100	Limit value 1
2	HYST1	I16	ENG	10	Hysteresis for limit 1
3	HYS1	B1	ENG	0	Hysteresis band 1: 0/1 = lower/upper
4	GRZ2	I16	ENG	980	Limit value 2
5	HYST2	I16	ENG	10	Hysteresis for limit 2
6	HYS2	B1	ENG	0	Hysteresis band 2: 0/1 = lower/upper
7	XIST	Source	ENG	AIN,1,XIST	Address of actual value
8	GRE1	B1	RT	0	Limit infringement 1: 0/1 = no/yes
9	GRE2	B1	RT	0	Limit infringement 2: 0/1 = no/yes
10	X	I16	RT	0	Actual value

GRZ1 and GRZ2 = Limit-Values
Hyst1 and Hyst2 = Hysteresis
XIST = Source for the analogue input

See the BRAUMAT-Manual Blocks S7, AIN for more Information

10 AOUT (Analogue Output Values)

Analogue outputs are handled by the **AOUT**-object of BRAUMAT.
Max. 255 AOUTs per PCU are possible
All AOUT-Modules have a fix output-address, but it can be modified if needed!

Example: *AOUT 1:* *PQW 512*
AOUT 2: *PQW 514*

10.1 Parameterization

You go to the Parameterization and select AOT.
You setup the dataset-name and the rest can be like that:

Braumat V6.0 {Area1} Parametrization - AOUT.1 - Flow

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	SOLL	Source	ENG	PID,1,Y	Source of physical target value
2	DigX_ANF	I16	ENG	0	digital start value
3	DigX_END	I16	ENG	27648	digital end value
4	XANF	I16	ENG	0	physical start value
5	XEND	I16	ENG	1000	physical end value
6	RAMP_PHYS	I16	ENG	100	Max. change per sec. of physical value
7	DigValType	Byte	ENG	0	AO format: 0=S7, 5=S5, 6=S5(sign+abs)
8	SIMU	B1	ENG	0	Simulation: 0/1 = no/yes
9	SPER	B1	ENG	0	Output disable: 0/1 = no/yes
10	LIMIT	B1	ENG	1	Limit XANF<=Y<=XEND
11	ISOLL	I16	ENG	0	Target value

SOLL = “PID 1,Y” corresponds to the source of the value
(in this case the PID Output)
XANF = 0 corresponds to the start of the range of the source
XEND = 1000 corresponds to the end of the range of the source
RAMp_PHYS = 100 corresponds to the max change of the output per second
(to make it more smoothly).

See the BRAUMAT-Manual Blocks S7, AOUT for more Information

11 TIMER (for Digital Inputs)

Digital inputs like a full- or empty-signals are handled by the **TIMER**-object of BRAUMAT.
Max. 1024 TIMERs per PCU are possible (two groups TIMER_1 and TIMER_2)
They have not a fixed input-addresses and must be transferred manually.

11.1 Transfer inputs to a timer- instance

In Order to have a faceplate for the inputs and a simulation, it is common to transfer all inputs (I 0.0 ….) except ICM Inputs (I 64.0 ….) to timer_1-input-flags (F 1240.0…).
Additionally you can define a delay for the Input.

Example: Transfer of inputs in a block (e.g. in FB1220)

L ID 0
T MD 1240
L ID 4
T MD 1244

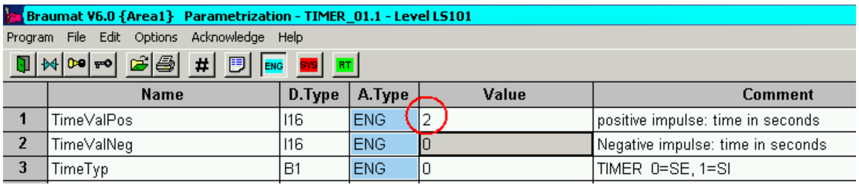
In the Program you need to use the Output of the Timer 1 (positive Signal) that means A F 1304.0 instead of A I 0.0. In this case you can simulate the Input from the process-image.

You can copy the list of the Inputs to file \Windcs\Pcu.001\TEXTE\ timer_01.txt in order to have the right names in the BRAUMAT-Process-Image.

Timer 1	*Start*	*F 1240.0….*
Timer 1	*Output pos.*	*F 1304.0….*
Timer 1	*Output neg.*	*F 1368.0….*
Timer 2	*Start*	*F 1432.0...*
Timer 2	*Output pos.*	*F 1496.0...*
Timer 2	*Output neg.*	*F 1560.0...*

11.2 Parameterization of a TIMER

You go to the Parameterization and select TIMER1-2.
You setup the dataset-name and the rest can be like that:



	Name	D.Type	A.Type	Value	Comment
1	TimeValPos	I16	ENG	2	positive impulse: time in seconds
2	TimeValNeg	I16	ENG	0	Negative impulse: time in seconds
3	TimeTyp	B1	ENG	0	TIMER 0=SE, 1=SI

TimeValPos = 2 is the delay on in second
TimeValNeg = 0 is the delay off in second
TimeTyp = Delay on (=0) or Impulse (=1)

12 PIDs (proportional – integral – derivative controller)

The PID - block contains all necessary functions for max. of 64 controllers per PCU.

The controller is suitable for:

- Fixed value control
- Cascade control
- Ratio control
- Hardware back-up control

12.1 Global flags of PID controllers

The flags for activating the “Follow-up” value are:

Interface-Flags:	PID	1	2
	F	*968.0*	968.1

12.2 Parameterization the PIDs

You go to the Parameterization and select PID.
You setup the dataset-name and the rest can be like that:

Braumat V6.0 {Area1} Parametrization - PID.1 - PID 1

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	Y	I16	RT	0	Manipulated variable
2	KP	I16	ENG	200	P-gain (S5:0.00-2.55, S7:0.00-327.67)
3	TN	I16	ENG	10	Integration factor=TA/TN
4	TV	I16	ENG	0	Derivative action factor=Tv/TA
5	A/H	B1	ENG	0	Operating mode: 0/1 = Auto/Manual
6	E/I	B1	ENG	0	Target value: 0/1 = external/internal
7	W	I16	RT	0	Effective target value
8	XIST	Source	ENG	AIN,1,XIST	Source for actual value
9	WEXT	Source	ENG	DFM1,2,SOLL	Source for external target value
10	Z	Source	ENG	#	Source error variable
11	YNF	Source	ENG	FIXV,1,ANA	Source for follow-up value
12	XD	I16	RT	0	Control error
13	XANF	I16	ENG	0	Start limit for XIST,WEXT,W
14	XEND	I16	ENG	1000	End limit for XIST,WEXT,W
15	YU	I16	ENG	0	Lower limit for manipulated variable Y
16	YO	I16	ENG	1000	Upper limit for manipulated variable Y
17	TEILANL	Byte	ENG	0	Assigned unit
18	YN	B1	ENG	0	YN flag
19	ART1	I16	ENG	0	Controller type 1
20	ART2	I16	ENG	0	Controller type 2
21	FOLG	I16	ENG	0	Number of the follow-up controller
22	TA	I16	ENG	1	Scan time in seconds (0=Locked)
23	TOB	I16	ENG	0	Dead band
24	WIED	B1	ENG	0	Restart: 0/1=un./manual
25	REV	B1	ENG	0	Reversing mode: 0/1 = No/Yes
26	XTR	B1	ENG	0	X-Tracking: 0/1 = No/Yes
27	X	I16	RT	0	Actual value
28	Wex	I16	RT	0	External target value

XIST =	source of the actual-value
WEXT =	source of the external target value (mostly coming from a DFM)
XANF,XEND =	range of the actual value
KP, TN, TV =	control - parameters for optimization.

See the BRAUMAT-Manual Blocks S7, PID for more Information

12.3 Special use cases for the PIDs

12.3.1 Tracking the PID- controller – output Y

In order to get the Output of the PID to “0“ when not in use (Pump, Steam off etc), do the following:
Using the YNF- flag (M 968.0) , the manipulated variable Y will be tracked to the follow-up value YNF in the Dataset.

- Go to the parameterization and link the input YNF with a fixed value (for example FIXV 1).
- Set the “FIXV 1” - value to the requested value (f.e. “0”) in the parameterization.
- Link the Output of the Pump or the Steam-valve to the YNF-flag to have it off if not needed.

12.3.1.1 Programming the PIDs in STEP7

Example: When the pump is not running, set the Output of PID to “0”

AN I 64.0 // Input ICM x
= F 968.0 // PID-YNF-Flag

12.3.1.2 Parameterization of a PID

Braumat V6.0 {Area1} Parametrization - PID.1 - PID 1

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	Y	I16	RT	0	Manipulated variable
2	KP	I16	ENG	200	P-gain (S5:0.00-2.55, S7:0.00-327.6
3	TN	I16	ENG	0	Integration factor=TA/TN
4	TV	I16	ENG	0	Derivative action factor=TwTA
5	A/H	B1	ENG	0	Operating mode: 0/1 = Auto/Manual
6	E/I	B1	ENG	0	Target value: 0/1 = external/internal
7	W	I16	RT	0	Effective target value
8	XIST	Source	ENG	FIXV,1,ANA	Source for actual value
9	WEXT	Source	ENG	FIXV,1,ANA	Source for external target value
10	Z	Source	ENG	#	Source error variable
11	YNF	Source	ENG	FIXV,1,ANA	Source for follow-up value

YNF = source of a fixed value

- PID-controller should be reset if not used by applying “0” to the YNF-Input of the controller and setting the YNF-Flag M 968.0…

12.3.2 Switch between two different setpoints at the PID (using MULT)

The multifunction block has 2 inputs and one output. Each block occupies one flag bit, which can have various meanings depending on its function. One of this (ART=0) realize a selector function.

- Go to the parameterization of the PID and link the Input WEXT with a MULT (for example MULT 1).
- Set two fixed value to the requested values for the MULT 1.

12.3.2.1 Programming the PIDs in STEP7

Example: If the Level is below x hl, reduce thes setpoint of the controller

AN F 760.0 // Level by DFM 1.1
= F 952.0 // MULT 1 interface-Flag

12.3.2.2 Parameterization of a PID

Braumat V6.0 {Area1} Parametrization - PID.1 - PID 1

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	Y	I16	RT	0	Manipulated variable
2	KP	I16	ENG	200	P-gain (S5:0.00-2.55, S7:0.00-327.67)
3	TN	I16	ENG	10	Integration factor=TA/TN
4	TV	I16	ENG	0	Derivative action factor=Tv/TA
5	A/H	B1	ENG	0	Operating mode: 0/1 = Auto/Manual
6	E/I	B1	ENG	0	Target value: 0/1 = external/internal
7	W	I16	RT	0	Effective target value
8	XIST	Source	ENG	AIN,1,XIST	Source for actual value
9	WEXT	Source	ENG	MULT,1,Y	Source for external target value

WEXT = source of the external target value

Braumat V6.0 {Area1} Parametrization - MULT.1 - MULT 1

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	Y	I16	RT	0	Output value
2	X0	Source	ENG	DFM1,1,SOLL	1st input value
3	X1	Source	ENG	DFM1,2,SOLL	2nd input value
4	ART	I16	ENG	0	Function type (S5:0-11, S7:0-8)
5	K	I16	ENG	0	Hysteresis

X0, X1 = source of two possible input values

13 Manual groups (HZUO)

You can define manual groups, in order to switch only one part of the plant to manual, for example one Unit. It could makes sense to use the units-number as the manual group-number e.g. “Unit 10” works together with “Manual Group 10”.

13.1 Programming the manual groups in STEP7

You can modify and check the flags in the program, e.g. in the user-FB 1220-1225

Example: Check the manual group status for number 10

```
A      F 713.1      // Manual group 10
SPB    MANU         // jump to the manual modus
```

The flags for the “Manual groups” starts at *F 712.0*

13.2 Parameterization of manual groups

There is no special object for the manual group inside the parameterization. The manual-group-number is used as an attribute in the parameterization of the ICM- and Unit-objects.

Braumat V6.0 {Training} Parametrization - ICM1.1 - SILO 1

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	TEILANL	I8	ENG	1	Assigned unit
2	HZUO	I8	ENG	1	0 none, 1..64 manual group, >64 MANUAL=1
3	TYP	I8	ENG	53	Type 8..13,16..21,32..38,48..53,128=locked

Braumat V6.0 {Training} Parametrization - Sequences.1 - MALT HANDLING

Program File Edit Options Acknowledge Help

	Name	D.Type	A.Type	Value	Comment
1	Enable	B1	RT	1	Permanent condition
2	Man_Moded	B1	RT	0	Manual / Auto mode
3	NewStep	I16	RT	0	Step new
4	Step	I16	RT	0	Step old
5	ManGroup	I8	ENG	0	0 none, 1..64 manual group, >64 MANUAL=1
6	BA_Year	I8	RT	12	Year for order no./batch no.

14 Other interface blocks

Beside the Unit - FBs, EOP - FCs and ICM – Interlock – FBs BRAUMAT provides a set of other interface – blocks.

14.1 Reserved user - function – blocks

You can implement your individual programming in the following blocks.

User-Program interface Begin OB1	*FB 1220*
User-Program interface End OB1	*FB 1221*
User-Program interface Begin 100ms OB	*FB 1222*
User-Program interface End 100ms OB	*FB 1223*
User-Program interface 100ms OB	*FB 1224*
User-Program interface 1s OB	*FB 1225*

Use FB1224 for fast Process-reactions (100 ms) and FB1225 for slow Process-reactions (1 s)

When the CPU had been switched off and on, these block will be called once.

Cold restart user interface	*FB 1200*
Warm restart user interface	*FB 1201*

14.2 Free blocks for other user-programming

In here you can call other FC, FB to do more user-programming

User-FC's	*FC300-499, FC2051…*
User-FB's	*FB200-500, FB1234….*
User-DB's	*DB2020…*
User-Flag	*Flag 1.0 .. 95.7, Flag 2050….*

15 Hints

- Avoid Set and Reset in Programming. It is not always save if you reset the sequence for example. Better use = -Orders
- Last Network in the EOP should be only for Next-Step-condition. Keep as simple as possible, that the Operator can understand.
- All Outputs in the Steps has to start with "AN M 102.3" to have a Reset at the end.

- In the Interlocks:should be Programmed only protection of Human and Machine. Operators don't like too much interlocks!

- You can have good PID regulation-results, if you start for x seconds the PID in the YNF-Mode with a certain value and then switch to Automatic.

- Use for Setpoints outside the recipe-system to be changed by the User the Special values. These can be used in Paramerization as Inputs for other modules as well. Avoid using FIXV (fixed-values) for this.

- For counting pulses, use FB 1222 or FB 1224. Minimum Pulse-length should be 100 ms Show pulse also in the screen.

- CPU goes to Stop, why? If it comes from a sequence, switch number of sequences in the DB725.DBW8 to "0" and start. Then switch the sequence on one by one (Sequence-Nr 1,2,3..)

- You can also use Timer from T 96 on if you set the number of INKU in the Paramerization to "0". Or you can use Timer from T 128 on if you set the number of 3PKTin the Paramerization to "0".

- Reset the Alarms if necessary at the start of the sequence, to have the Alarm at the right time

- Reset Manual-Group at the start of the sequence and set PID's to external and Auto.
  like this: U   DB725.DBX 101.1   //"SEQ".u.STATUS.boSeqStrtImp
  R   "PID".au[x].boA_H   //reset your PID's used in that Unit
  R   "PID".au[x].boE_I

- You can set a bookmark by writing any flag in the software with "A M 9.7" ; = M 9.7 and you find it later with cross-reference.

- You can start in the graphical recipe also a sequence in another PCU.

- Same Decoder can be used several times, if you check for the decoder in the Unit-FB and transfer the result to a flag.

- Avoid changing the Steps and synchronizations of a running Production-Recipe. It will let the synchronizations block and you have to release them manually. Changing Setpoints is ok.

- By the way, you can process the Symbolic-List in Excel if you export it to a .seq-File and drag and drop to Excel. Insert a fist line and input x in the first column, to avoid to be removed by saving.

-

15.1 Multi-Engineering

works nice with Braumat on one PCU. This helps us to organize it

- Use the Excel-Engineering-Sheet (d:\WINDCS\EXCEL\Param_PCU001_BH1.xls) to reserve DFM's, FIXV, MULT, MEKO etc. for each Unit.
- Also reserve Timer (SE_Timer for Inputs) for each part.
- Reserve Special-values for Setpoints and Timer-Values for each Part within a PCU. All Operator-Settings outside the recipe should be here, not in FIXV! Here you have a comment for each Setpoint etc. You can copy the text of Special values at the end in the Symbolic-List of Simatic-Manager.
- Reserve EOP's (Steps) for each Unit.
- We use one DB for each Unit for all the Flags, Integer etc. You avoid by this using the Symbolic-Table (which is global).
- Interlock-FB'S FB1226… are global FB. You can upload these from the PLC prior to change.
- If needed we changed the Symbolic-File direct in the Master-Project.
- Everybody changes direct at the system online! Then all have the same database. It is very difficult to merge recipe from different Sources!!
- Clients change Data always on both Server, so don't worry about Picture-Changes and recipe-changes.
- We synchronized our changes every evening in the Master-Project, and everybody took this the next day as basic for programming.
- In this way 3 person had been able to work in one PCU at the same time! Trigger for Status can not be increased unfortunately in Simatic-MAnager. But you can see the Status also with Braumat (the flags direct or the blocks!).

15.2 Following Interlocks are mandatory

- Pumps must have an open way (Inlet is more important than Outlet), to avoid running dry.
- Steam-valves must have liquid in the vessel or in the pipe to avoid overheating.
- Close inlet-valve if the tank is full
- Door-Switches, safety switches interlock all relevant elements that can harm.
- Product and CIP is interlocked?
- If you pump to a vessel, check that there is no CIP and no Production running.

16 Route Control System (RCS)

The Route-Control-System is used to define ways for example in the fermentation-cellar for the transfer of beer from one tank to the other one.

In the RCS-Application you import **ICM** first
Define a PCU and Import ICM by the Project in Database, Import ICM

Then you Input the **Sensor-Elements** manually or by Exporting to a csv-File and copy/Paste the Name from the Timers.

Parameter-Elements are set points static or dynamic (from DFM) for calculation of Pipe-content Quantity etc.

Link-element: Material for one pipe or Tank

Mode-Table: Table with max. 32 Action-Flags for switch on the way, then the Pump etc.

Master for: max Routes active at the same time

For using the RCS, you need to Input in Parameterization
Fifos 1 and 3: Type 30, 31, 32, 33, 34, 36

```
 CALL  "RC_CALL"  //FC 820
     TR_CE_ICM:=TRUE                  // Benutze ESGs
```

In the FB1220 and FB1225

Also in Parameterization you can adapt the project in RC_CNF

Step7 Interface-Programming for RCS

16.1 RCS-Programming in Unit-FB's

Step7 – interface - programming for RCS in the **Unit-FB's FB 100x.**
Put the following code for RCS into the network "Action after GOP was running":

```
// no way found
      A(
      L      DB725.DBW  186          // Route – no., "SEQ".u.iROUT_ID
      L      0
      >I
      )
      A      DB725.DBX  410.1        // "SEQ".uRCS.QREQ
      A      DB725.DBX  342.1        // "SEQ".uRCS.REQ
      AN     DB725.DBX  110.2        // "SEQ".u.CTRL.boRestart
      S      DB725.DBX  110.5        // "SEQ".u.CTRL.boPaused

// general RCS fault --> stop recipe
      A      DB725.DBX  434.2        // "SEQ".uRCS.QERR
```

```
        A       DB725.DBX  342.1          // "SEQ".uRCS.REQ
        AN      DB725.DBX  342.6          // "SEQ".uRCS.ACK
        S       DB725.DBX  110.5          // "SEQ".u.CTRL.boPaused
```

16.2 RCS-Programming in EOP-FC's

Step7 – interface - programming for RCS in the **EOP-Blocks FC1001……**

16.2.1 RCS_START

Implement the following networks for RCS in the Step: RCS_START

```
// All Modes off
        L       0
        T       DB725.DBD  400
// Mode table
        L       2
        T       DB725.DBD  396            //"SEQ".uRCS.MODE_TBL
// Way-ID
        L       2                         // here you Input the Way-Number
        T       DB725.DBW  186            //"SEQ".u.iROUT_ID
// Way Properties:
        L       302
        T       DB725.DBW  340            // "SEQ".uRCS.FUNC_ID        // Func_Id
// If GOP running, then request On
        AN      DB725.DBX  111.4          //  "SEQ".u.CTRL.boRestarting
        AN      DB725.DBX  110.2          // "SEQ".u.CTRL.boRestart
        =       DB725.DBX  342.1          // "SEQ".uRCS.REQ
// If GOP restarting, then remain restarting for 1 cycle
        A       DB725.DBX  111.4          // "SEQ".u.CTRL.boRestarting
        R       DB725.DBX  111.4          // "SEQ".u.CTRL.boRestarting
        A       DB725.DBX  110.2          // "SEQ".u.CTRL.boRestart
        S       DB725.DBX  111.4          // "SEQ".u.CTRL.boRestarting
// GOP started
        O       DB725.DBX  111.4          // "SEQ".u.CTRL.boRestarting
        O       DB725.DBX  110.2          // "SEQ".u.CTRL.boRestart
        ON      DB725.DBX  342.1          // "SEQ".uRCS.REQ
        =       DB725.DBX  342.6          // "SEQ".uRCS.ACK
// Way found
        A(
        L       DB725.DBW  186            // "SEQ".u.iROUT_ID
        L       0
        >I
        )
        A(
        L       DB725.DBW  408            // "SEQ".uRCS.QREQ_RC
        L       4                         // according ID
        ==I
        )
        =       DB725.DBX  342.2          // "SEQ".uRCS.RON
// Provide more functions
        SET
```

```
        R       DB725.DBX  342.5          // "SEQ".uRCS.SOLID
        R       DB725.DBX  342.4          // "SEQ".uRCS.IGN_ERR
        R       DB725.DBX  343.0          // "SEQ".uRCS.SET_MAT
        L       0
        T       DB725.DBD  352            // "SEQ".uRCS.VIA_1
        T       DB725.DBD  356            // "SEQ".uRCS.VIA_2
        T       DB725.DBD  360            // "SEQ".uRCS.VIA_3
        T       DB725.DBD  364            // "SEQ".uRCS.VIA_4
        T       DB725.DBD  368            // "SEQ".uRCS.VIA_5
        T       DB725.DBD  372            // "SEQ".uRCS.VIA_6
        T       DB725.DBD  376            // "SEQ".uRCS.VIA_7
        T       DB725.DBD  380            // "SEQ".uRCS.VIA_8
        T       DB725.DBD  384            // "SEQ".uRCS.VIA_9
        T       DB725.DBD  388            // "SEQ".uRCS.VIA_10
        T       DB725.DBD  344            // "SEQ".uRCS.MATERIAL
// load Way Source and Destination
        L       DB737.DBD 4570            // "DFM1".au[195].rSPVal, Setpoint for Source
        T       DB725.DBD  348            // "SEQ".uRCS.SOURCE
        T       DB737.DBD 4574            // "DFM1".au[195].rActVal

        L       DB737.DBD 4592            // "DFM1".au[196].rSPVal, Setpoint for Dest.
        T       DB725.DBD  392            // "SEQ".uRCS.DEST
        T       DB737.DBD 4596            // "DFM1".au[196].rActVal
// last Network Transition
        A       DB725.DBX  410.2          // "SEQ".uRCS.QON  Way is on
```

16.2.2 RCS_TRANSFER

Implement the following networks for RCS in the Step: RCS_TRANSFER 1

```
// All Modes off
        L       0                         // All Modes off
        T       DB725.DBD  400
// Request und Way On
        AN      "SEQU_ESto"
        =       DB725.DBX  342.1          // "SEQ".uRCS.REQ
        =       DB725.DBX  342.2          // "SEQ".uRCS.RON
        =       DB725.DBX  403.0          // "SEQ".uRCS.MODE_01 activate Way
        =       DB725.DBX  108.7          // "SEQ".u.CTRL.boTime_Rel_Req release Time
// switch on pump only if RCS is ok
        A       DB725.DBX  403.0          // "SEQ".uRCS.MODE_01
        A       DB725.DBX  417.0          // "SEQ".uRCS.QMODE_01
        =       F 12x.y                   //CA_Flag Pump etc
// transition
        A    DB725.DBX  417.0             // "SEQ".uRCS.QMODE_01 according mode on
```

17 More recommendations which are not mandatory

17.1.1.1 Standard Frame

This is a predefined standard file which is the basis for all graphical interfaces in the project. The Standard frame defines areas for screen navigation, standard applications, alarm controls, user login and Process Pictures as well.

Define a Standard-Frame-Size, depending from Screen-resolution f.e. 1640 x 820. Non Process Picture's Message refer to global messages of the Area.