Herman Automation

We boost your efficiency.

ABB AC450 translation to AC800M code by easy web-tool that saves you time

Ampl2m conversion tool was developed during real migration projects in order to simplify migration of ABB Advant Master AC400 family controllers. AC450, AC410, MP200, AC110, AC70, AC80, APC are supported.

Manual translation of AC450 programs to AC800M is very challenging. Manual translation of one Controller may take 6 months or longer. Challenges of AC450 translation are:

- Control application is typically very complex containing thousands of pages of logic
- Control programs may be written in tricky way containing undocumented so-called “hidden terminals” of database elements, which makes the logic difficult to understand hence difficult to translate
- Programs using special libraries like RMC, RPC, APC, QCS/1190
- Programs written deliberately in tricky way, e.g. logic dependent on execution order and timing of DB elements reading and writing
- Logic containing DB terminals, which are not available in AC800M, like :SELECTED
- Sequence part translation to SFC requires extensive experience to prevent bad surprises during commissioning

Ampl2m tool cannot accomplish 100% complete AC450 conversion, but it can significantly reduce engineering efforts and prevents human errors. Conversion tool is able to save about 60-95% of working hours, based on experience. Remaining unconverted signals and functions, displayed by red color in AC800M code, should be completed manually.

It is really worth to use the conversion tool in order to reduce all that exhaustive manual migration job, based on experience of the author in several projects already commissioned.

Core Benefits

- Automated conversion can save hundreds of working hours per one Advant CPU
- Conversion tool is easy-to-use and available on-line 24 hours every day
- Prevents human errors
- Conversion process is interactive thus under your control
- Pulp & Paper Library –or– Standard AC800M libraries –or– User libraries may be selected for conversion
- Converter utilizes ready-to-use SW solutions verified during successful projects and long-term experience in AC450 & AC800M programming
- Cost-saving in terms of tag count
- AC800M code optimization functions improving code readability

How does it work?

Conversion tool is built as database application hosted at Virtual private server (VPS) . It is available as web service at **ampl2m.com** and accessible through simple web interface ready for instant conversion. Just log in, upload source files and press Analyze and Convert button. Converted programs are available for download in few minutes.

Downloaded text files contain all Control modules, variables, data types and constants ready for import into 800xA or Compact Control Builder. Import is performed by means of short VB script using ABB Open interface of Control Builder.

Security, Privacy

We are committed to protect your data. We take the protection of your personal data very seriously. We are using SSL encrypted connection (https:), so third parties do not have access during transmission of data between your browser and our Virtual Private Server (VPS). You can recognize an encrypted connection in your browser's address line when it changes from http:// to https:// and the lock icon is displayed in your browser's address bar.

Our VPS is located in the Data center of VPS Provider in Germany secured by DDoS protection and Firewall. Your Input and Outputs files are stored inside the database in our VPS. External connection from internet to our database is disabled, therefore the database can be accessed by Conversion service only . Remote access to VPS is secured by SSL Private Key thus restricted to Herman-Automation only.

Herman-Automation does not share with third-party nor sell to third-party Your personal data or Your files. Read our Terms&Conditions for further information.

What is the efficiency of Ampl2m conversion tool ?
How much manual work is needed after automated conversion ?

Automatic conversion can save up to 60-95% of manual AC450 translation, depends on how original code is written. If original control application is built using MOTCON, VALVECON, PIDCON, SEQ mainly, then automated conversion covers almost all engineering efforts.

According opinion of many ABB specialists complete AC450 to AC800M automated conversion is not possible due to significant difference with application handling between AC400 and AC800M. Manual post-conversion job and experience are important part of the conversion process.

Amount of necessary manual programming after automated conversion depends on the way how original code is written:

- Automated conversion produces nearly working AC800M code if original logic uses standard features of AC450 function blocks and DB elements – in case standard MOTCON, VALVECON, PIDCON, SEQ are used
- On other hand converted code needs manual fixes proportional to amount of GENUSD, GENCON, GENBIN, MMCX used and on how sophistically and tricky is original code written.

In our experience, there is enough room for partial automated conversion which can significantly reduce engineering hours and prevent typos, mistakes due to inattention or logic errors. **Manual completion after automated AC400 conversion may take 2-6 weeks, depends on experience. That manual post-conversion job is much easier than at least half-a-year-long manual reprogramming of AC400 to AC800M.**

You can send us AAX, BAX files for assessment how many working hours will be needed in your project for manual fixes in converted code.

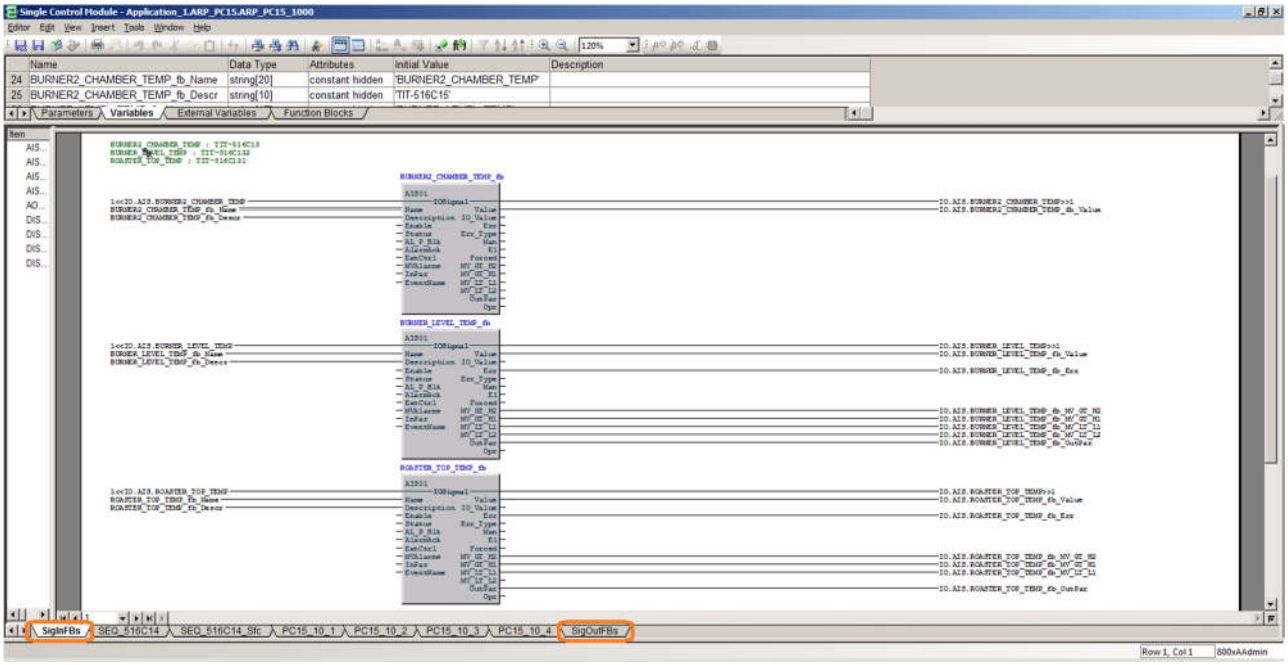
Features

- Input files required for automatic conversion: *.AAX , DBT (Advant controller database export file), *.TCS in case AAX files contain Type circuits. Note: BAX file is not required hence it secures AAX files cannot be opened in Function Chart Builder by other person.
- Output TXT file contains translated all global variables, constants and Single Control Modules (PC programs) ready to import into 800xA / CCB. Typically one output TXT file contains one translated PC program.
- Advant AMPL code is converted into Single control modules(SCM) using FBD or SFC language.
- **IO function blocks (represent DB blocks AIS, AIC, DIS, DIC, AOS, AOC, DOS, DOC)** are created in each SCM if their VALUE is red or written there.

  There is built-in logic, which decides for the most appropriate location of each particular IO function block in Single control modules (PC programs), according following rules in descending priority:

  - The fastest program cycle is selected with the connection to the particular DB / IO element
  - IO function block VALUE or CALC_VAL is written
  - IO function block VALUE or other terminals are red the most times in particular PC program
  - These AIS and DIS, which are not connected in any PC program, their IO function blocks are created in dummy PC99 generated by conversion tool. That dummy PC99 shall be converted as the last one. The most likely AIS and DIS collected in dummy PC99 are used as independent measurements just displayed in HMI graphics.

  Local distribution of IO function blocks in SCMs is very helpful for easy navigation from HMI / Alarm list / HW IO cards to the logic. Example of how IO function blocks are created at the first and the last tab of each SCM:



  - RealIO and BoolIO variables are created locally in the same SCM along with corresponding IO function block in case of Local preference, or globally in case of global preference. IO variables are furnished with original IO name, Range, Unit and Fraction in variable description.

- Name and Description texts are created as local “constant hidden” strings in SCM (by default, as per example above) or as project constants organized in folders of constants like cNames.<Application name>.<Const name>. The same is applied for Names and Descriptions of Motcons, Valvecons, Pidcons, Manstns and SEQ function blocks...

  Name and Description constants are uploaded automatically to the Control Builder during importing of converted code.

- Extensions are being added in the end of IO FB names in order to prevent collision between FB name and RealIO/BoolIO variable name.

  Name extension of IO FB is adjustable in the converter’s user settings. Default extension is “_fb”. If extension setting is preset to “_#” then particular type of IO function block is used as extension, e.g. <DB name>_AIS , <DB name>_DOC ,…

- IO values are referenced in the logic in 2 ways:

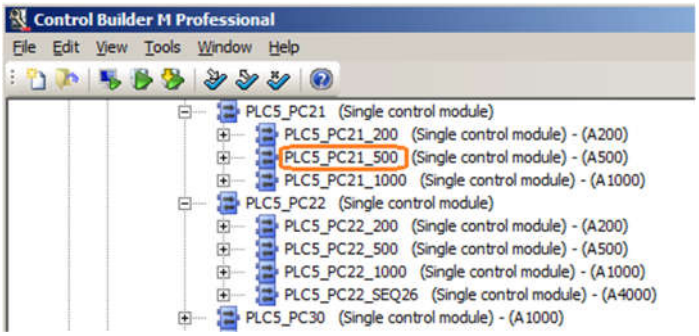
  - Locally by <IO FB name>.<terminal> within the same SCM where IO FB is located

  - by global variables connected to the terminals of IO FB, if IO value is used also in other SCM (s). See chapter “Preference for local/global variables” below.

- Original Descriptions of IO function blocks are also added in Page comment in which particular IO FB is located and also in the remark above FB in process logic reading values from IO FB

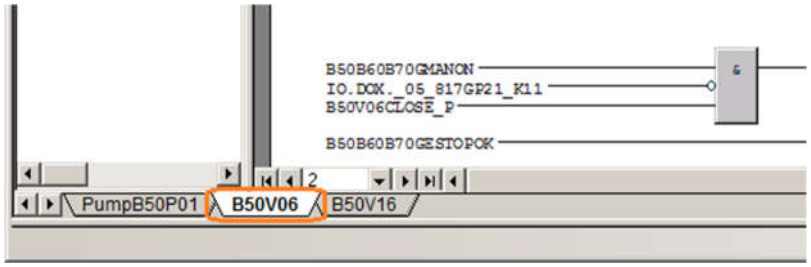
- If Local variable preference is preselected before conversion then RealIO / BoolIO variables (to be connected to IO cards) are created locally in the same SCM in which particular IO FB is created.

- **Code distribution** can be adjusted by setting appropriate names of target Control Modules and code tabs. If the same name is set for several code blocks (FUNCMs), their code is merged into one code tab. This approach can decrease AC800M CPU load significantly, especially if there are a lot of small FUNCM/SLAVEM code blocks in AC450.



- Control Module names can be modified before conversion. Control modules can be split/merged by setting new/other existing CM name instead of CM name preset by Analyze function.

Analyze function presets Tab names by default as original PC addresses of FUNCMs like PC22_3_2_6. Modify Tab names according particular control function or plant part. Merge tabs if possible before conversion by using the same Tab name for several original FUNCM code blocks.



- **Preference for local / global variables.**

  In case Local variables preference is set, global variables are created only if particular variable is used in more then one Control module, mainly in case of communication between SCM of different cyclicity :

  This preference is applied for named connections in the logic and for connection between IO function blocks and the logic. Local IO signals are referenced by <IO FB instance name>.<terminal> .

  Local or global preference - what is the best option?

  Global variables preference is handy because all variables are accessible across all converted PC programs.

  On other hand, amount of global variables is limited within one Application (~65536). Unfortunately Control Builder complains about maximal limit is exceeded just on download to the real AC800M CPU. This problem may be discovered too late if CPU is not available during programming phase of the project.

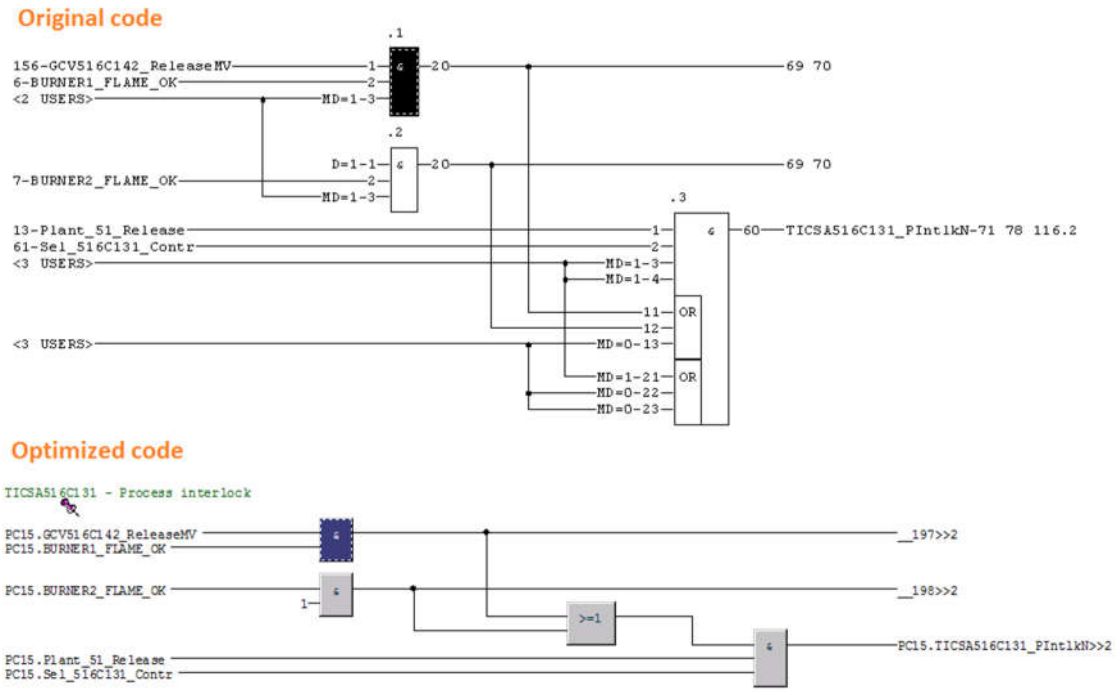
  Therefore the safest option is Local variable preference especially if control logic is complex – let say in case of more than 20 PC programs per one CPU.

- **Time entries** (like D=1:0:0) can be converted as time constants (cTimes._1h) or coldretain time variables if adjustment of time entries is expected.

  All time constants used are listed in the end of output TXT file and they are automatically added as project constants during importing converted code into 800xA. The same applies for String constants.

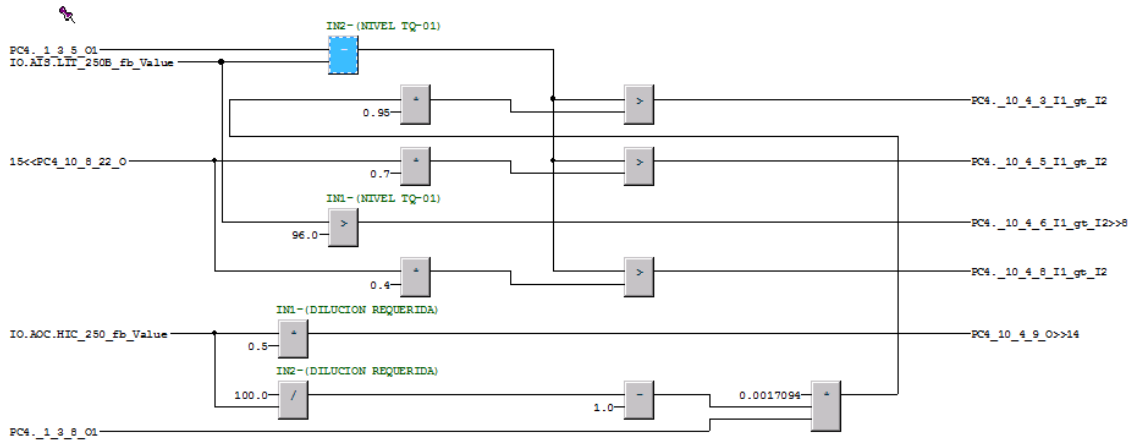
- **Code optimization on/off.** Code optimization improves readability of translated code.

  If code optimization is on, spare inputs of AND, OR elements are omitted, or whole spare AND, OR elements and zero time Timers are bypassed. COMP+OR are replaced by GE, LE functions. Example:



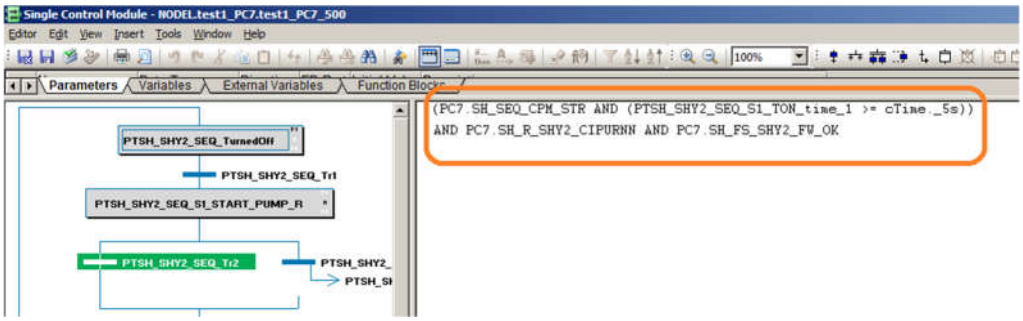
- **Symbolic names translation.** Original symbolic names at PC element outputs are used as names of new variables connected at converted function blocks outputs.
    - If original symbolic name is enclosed in parentheses, symbolic name may contain even special characters. In that case conversion tool converts special characters as abbreviations, e.g. “<” is replaced by “_lt_” , “%” is replaced by “perc”...
    - In case there is no symbolic name available for the particular PC element output, following rules are applied:
        - In case of objects (Motcon, Valvecon, Pidcon, Manstn, AIS, DIS,...) , output variable name is created as <FB name>_<terminal name>
        - if output variable is connected in other Single Control Modules, then global variable is created like
          <PC prg.>.<remaining part of original PC address>_<PC element output terminal>
          e.g. PC15._1_6_O
        - if output variable is connected to the neighbour code tabs within one Single Control Module, then named variable is created like
          < original PC address>_<PC element output terminal>
          e.g. PC15_1_6_O
        - if output variable is connected inside particular Code Tab only, then variable name begins with 2 underline characters – these variables are not listed in variable list of Control Module
    - **In case symbolic names are lost** in uploaded .AA files from Advant controller , it is possible to recover original symbolic names by script. In that case ask customer for older AAX files, which contain symbolic names still. Send us these AAX files and we will extract symbolic names and transfer them into your latest AAX files. This automatic symbolic names recovery has been used successfully in the real project already.

Example of Variable naming if original symbolic names are not available:



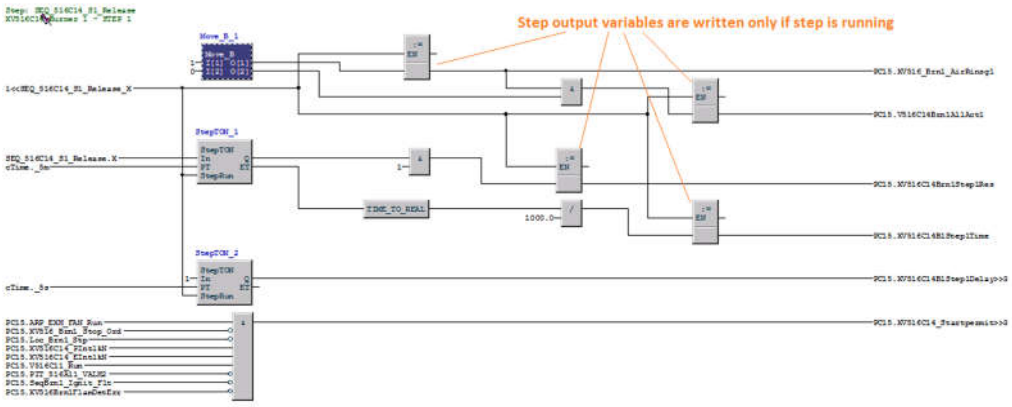
- **Selection of STEP transition variable:**
    - single bool variable
    - whole logic expression.

    What can be helpful for more informative HMI in SFC Viewer (available only if whole STEP logic is converted into SFC only).

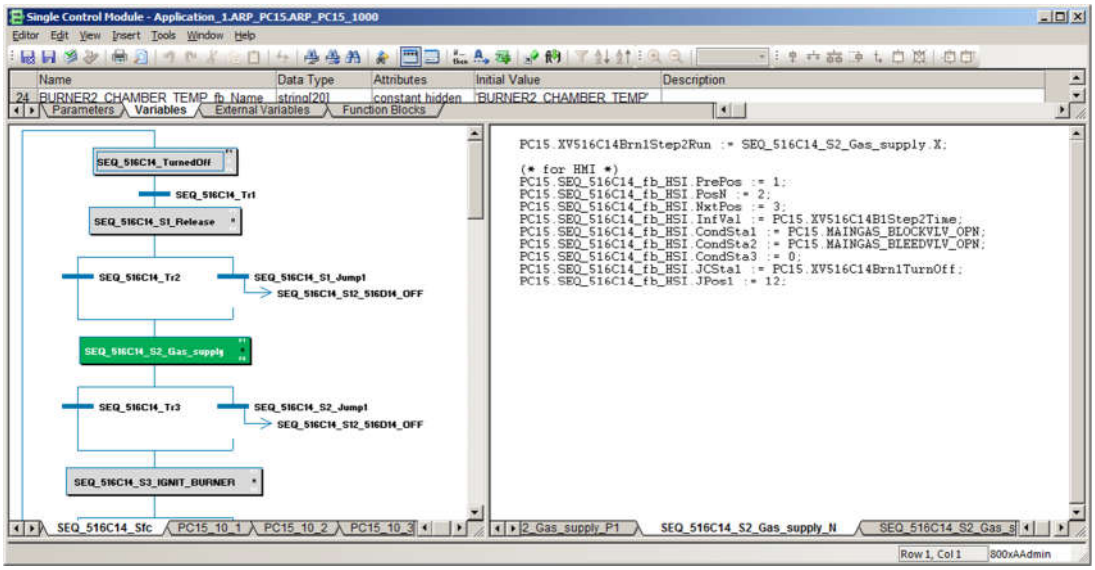


- **SEQ conversion:**
    - Whole STEP logic is converted into SFC as by default. This is the best option if Steps contain just one page of simple logic like few timers and MOVE sending commands to the valves, motors and PIDs.
    - In case STEP logic is complex, for better readability it is useful to convert STEP logic into FBD code located one tab before SFC. In this case SFC is used as a „SEQ skeleton“ only.

Step code example, if FBD is selected in User settings for SEQ translation:

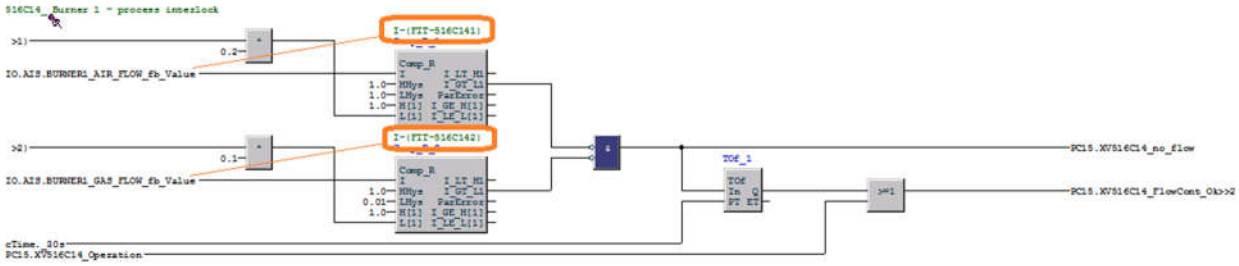


SFC skeleton example:

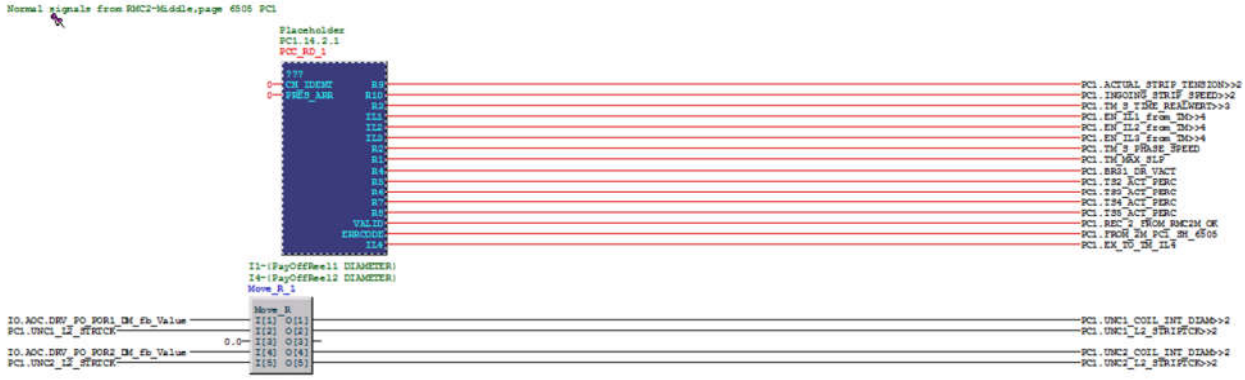


- **Number of 800xA Tags can be optimized** by means of user filters of IO signals which do not need faceplates. The aim of tag filters is to avoid using IO FBs for motor's MCC signals and limit switches of valves, which values are being displayed in faceplates of corresponding valves and motors already.

- **Description of connected IO signals** is added into the remark above function blocks.

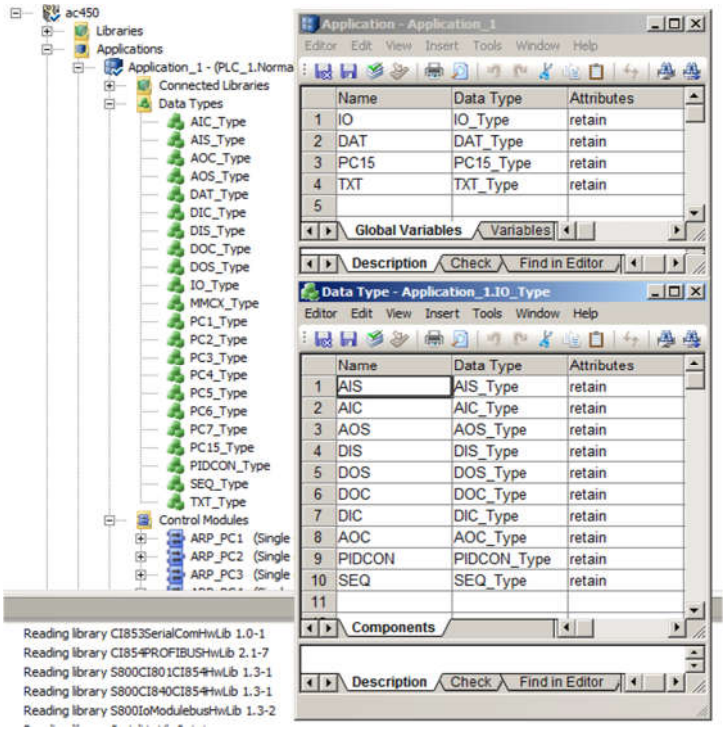


- In case of rarely used PC elements, which are not defined in conversion tool yet, they are replaced by red **placeholders**. At the same time all input / output logic connections are translated and maintained. Names of placeholders are added to the FunctionBlocks definition tab. Just new FB definition in User Lib remains to be done manually.



All unconverted function blocks are listed in the end of the output TXT file.

- **Data type structure** prepared by automated conversion is shown in the following screenshot. Data types are created in Control Builder during importing of TXT files, which contain Control modules and data types. During sequential importing of several TXT files (since 1 TXT file contains 1 PC program usually) variables are added into the existing data types imported before. No variables previously created are deleted during importing into Control Builder.
    - Description of every variable contains information where particular variable is written from – source SCM name and Tab name.
    - Unit is also placed in variable description.
    - Initial value is set for time and string constants and for numerical constants used multiply in the logic (MD=…)
    - All original int variables are converted as dint



- **Tag counter is built in**. Tag quantity is shown in the name of each output TXT file.
- **Type Circuits.** Conversion tool translates Type Circuits in case:
    - AAX files contain patterns like (* Begin of Type Circuit -and- (* End of Type Circuit
    - all TCS source files are available and uploaded at your Input files before conversion

    Type circuits are translated as Function blocks instances in FBD code. Note that Function block types should be prepared manually in the current version of the Conversion tool.

- **IO variables connection to IO cards.** CSV file (comma separated text) is created during importing of translated logic to Control Builder M. CSV file contains list of BoolIO, RealIO variables used in the converted logic including DB element original names, Path of IO variables, Analog ranges, units, fractions and channel inversion. CSV file data can be easily copy-pasted to the IO card editor. We cannot generate complete HW structure of IO cards since we don't know your new IO cards layout and IO signals distribution in IO cards.

Features in progress, coming soon

- Init code for Pulp&Paper IO function blocks, executed only once on startup. The aim of init code is to set signal ranges, H2,H1,L1,L2 alarm setpoints, AE configurations, PID controller constants,…
- Temporary code for simulation of motor MCC feedbacks and limit switches of valves – useful for simulation of the plant before and during FAT tests.

Libraries

PP Library (Pulp & Paper) is used in converted code as the first option (default). If PP Library is available and/or requested by customer it is advantageous for AC450 conversion because:

- PP function blocks are compatible with Advant controller's function blocks.
- PP HMI Faceplates have similar functionality as control objects at AS500 operator stations and nearly the same look and feel like 800xA+AC400connect HMI.

If PP Library is not available there is **another reliable and cost-saving solution by means of using Standard AC800M libraries + UserLib**. In this case FBs are used from BasicLib, ControlSimpleLib mostly. In addition there are 18 function blocks created in UserLib replacing AC450/APC FBs which are more complex hence not easily convertible by standard library FBs. UserLib FBs were tested thoroughly and used in the real projects already. UserLib is available for download after your first accomplished conversion. UserLib is password free hence open for modifications.

See how Libraries are utilized in the list of supported function blocks in the end of this document.

Testing of converted code

It is recommended to perform thorough tests of converted code in simulation mode before FAT. All motor MCC feedbacks and limit switches of valves should be simulated in AC800M while IO cards are disconnected. The aim of tests is to perform simulated start and stop of controlled plant. It is necessary to run all sequences and check if all particular motors, valves and PID controls perform as expected.

Conversion tool generates temporary simulation code in each Single Control Module in order to simulate Motor MCC feedbacks and limit switches of valves.

Conversion tool released version

Ampl2m conversion tool has been tested by 8 experts in ABB control systems. It has been used in 6 migration projects reported up to now. The tool has been used by the author in 3 projects including commissioning.

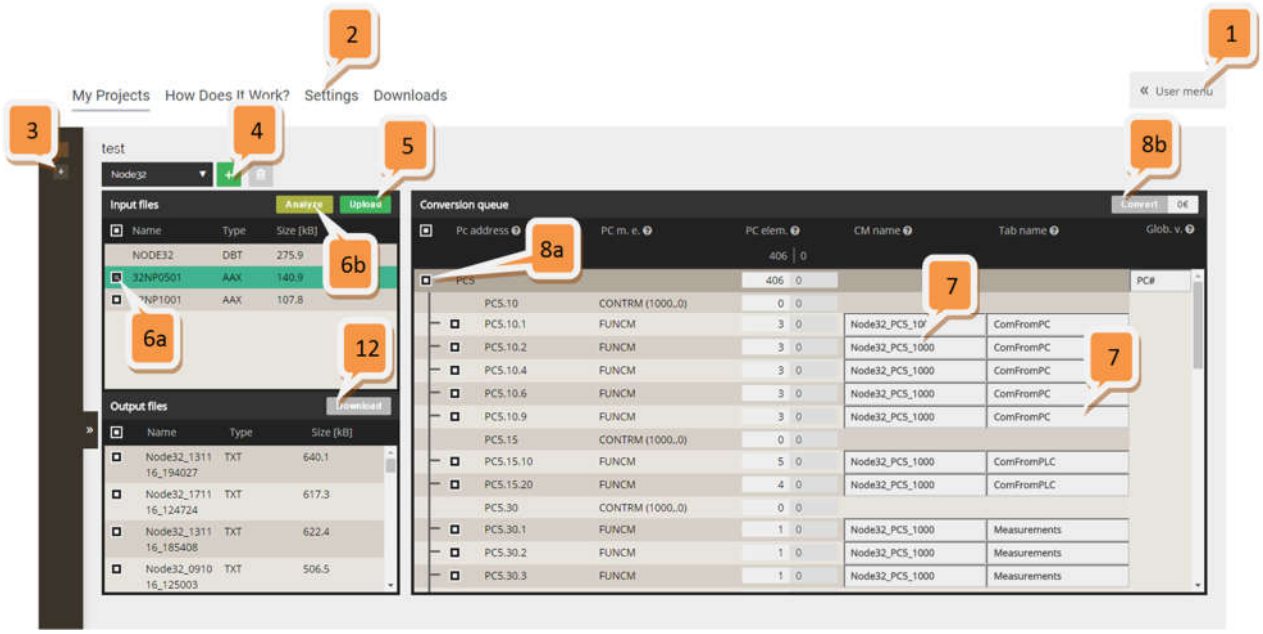
Ampl2m conversion tool development is in progress with the aim of increasing its efficiency. Meantime the conversion tool is ready to help in migration projects as solid stabile version. Any bug reported will be fixed ASAP. Punch lists or any idea for tool improvement are appreciated much. It is possible to adapt the conversion tool for your project specific needs or for other libraries, see “List of supported PC elements” chapter.

Guarantee of Service quality, Warranty

Herman-Automation warrants that, for any paid Conversion , for PC elements supported only, the Conversion will produce control logic equivalent to the original control logic from Input files (Equivalent logic means that converted logic is producing the same outputs as original logic if both control logics are working with the same inputs).

The warranty does not apply to the free Conversion.

Automated conversion Workflow



1. **Create your account** at Converter web site

2. Check and adjust your **preferred settings** e.g. select Libraries to be used, Page layout, local/globar preference, code optimization settings...
3. **Create new Project**
4. **Create new PLC.** PLC name must be unique within one Project. You can use PLC name from your Advant engineering PC FCB.
5. **Upload PC source files** *.AAX and DB export file *.DBT from FCB (use File -> Export DB section… and confirm default settings). Only one DBT files should be uploaded containing the export of the whole DB part. All AAX files should be uploaded prior to the first conversion of particular PLC.

6. **Select AAX file(s) for analysis and click Analyse button.** All AAX files can be selected and analyzed at once. Converter parses selected AAX files and calculates number of convertable function blocks.
7. **Check and adjust suggested Control Module names and Tab names** in Conversion queue table. It is highly recommended to change Tab names to appropriate names like plant part abbreviation. Merge several code blocks by means of entering the same Tab name. Rule of thumb: Less Tabs = lower AC800M CPU load and faster opening/saving of CM editor.
8. **Convert:** Select code blocks in Conversion queue table and Click Convert button. It is recommended to convert PC programs one by one (one output TXT file per one PC program). Do not convert only part of PC program because some variables may remain unconnected. Alternatively click on PC program header line in Conversion queue to fold/unfold PC prg.
   Convert automatically generated file LAST9901.AAX as the last one. That file contains all IO function blocks, which are not connected in PC programs.
9. **Select** limited conversion for free or full conversion if you would like to (demanded). This selection has no meaning meantime since conversion tool is free.
10. After selection of the full or demo conversion, popup "Conversion status" appears and displays the progress of conversion.
11. Conversion takes from several seconds to several minutes depending of AAX Advant code complexity
12. **Download converted text file**, which contains all converted Control Modules and data types in text XML format ready to import to your 800xA system.
13. **Import converted text file** to the 800xA system by means of VB script (available in Download section)
14. Repeat conversion any times you want for free with adjusted settings or Control modules / Tab names modified. All your previously modified setting are stored automatically at your web browser local data storage, so it will be available next time at Conversion queue table.
15. Define global variables using imported data types like IO,DAT, AIS, DIS, AIC, AOC,...

Post-conversion Phase 1

is defined as manual programming after automated Conversion, in order to fix all unconverted parts of Output files and to get converted programs ready to download to AC800M controller, including following activities:

- Visual checking of imported code page by page according original logic
- fixing all Placeholders in converted Output (Placeholder replacement by appropriate Function blok of AC800M)
- fixing all red connections between Function blocks. Converted code may contain red connections, be-cause special DB hidden terminals or terminals SELECTED were used in original logic.
- fixing all Code loop errors during compilation in Control Builder M
- assigning Tasks to Converted code in Control Builder M
- defining communication between other controllers as replacement of DS and DSP communication blocks
- Downloading project to the Softcontroller

Post-conversion Phase 2

is defined as configuration in Control Builder/Engineering workplace in Customer's 800xA system in order to:

- to set up Coldretain values into Pulp&Paper function blocks in Control Builder M online
- to build hardware definition of AC800M and all its hardware components
- to setup IO variables to IO cards, in Control Builder M
- to configure texts of Sequence steps in 800xA Engineering workplace
- to configure interlock texts of Valves and Motors in 800xA Engineering workplace
- to download converted code into AC800M CPU and to optimize CPU load and Tasks
- to perform FAT test with Customer / end-user
- Commissioning of converted code

Herman-Automation can take over responsibility for post-conversion activities and can provide Post-conversion Service including whole or part of Post-conversion activities defined above, according Purchase Order from Customer.

List of supported PC elements

Ampl2m conversion tool supports now almost 100 PC elements, which are the most frequently used in the control logic of Advant controllers. Conversion tool is improving after each project in which it was used thanks to requests and punch lists from users.

call_name	defined by Basic/User Libraries	defined by P&P Library
AND	Basic Lib	
OR	Basic Lib	
TON	Basic Lib	
TON-RET	Basic Lib	
MONO	Basic Lib	PP Lib
MOVE	User_Lib	PP Lib
MOVE-A	User_Lib	
OR-A	Basic Lib	
INV	Basic Lib	
STEP	Basic Lib	
CONV	Basic Lib	
SR	Basic Lib	
FUNCM	N/A	N/A
TRIGG	Basic Lib	
SR-AO	Basic Lib	
SW-C	Basic Lib / User_Lib (T)	PP Lib
TOFF	Basic Lib	
MUL	Basic Lib	
SUB	Basic Lib	
SR-OO	Basic Lib	
COMP-R	User_Lib	PP Lib
AND-O	Basic Lib	
DIV	Basic Lib	
BLOCK	Basic Lib	
INT	ControlSimpleLib	PP Lib
ADD	Basic Lib	
REG-RET	User_Lib	
COUNT	Basic Lib	PP Lib
CONV-BI	Basic Lib	PP Lib
LIM-N	Basic Lib	PP Lib
COMP-I	Basic Lib	PP Lib
REG	User_Lib	PP Lib / User_Lib
MUXA-I	User_Lib	PP Lib
THRESH-L	Basic Lib	PP Lib
COMP	Basic Lib	
OSC-B	Basic Lib	PP Lib
SR-AA	Basic Lib	
CONV-IB	Basic Lib	PP Lib
MUX-N	Basic Lib	PP Lib
MUX-MN	Basic Lib, User_Lib	PP Lib / User_Lib
COUNT-L	Basic Lib	PP Lib
SR-D	User_Lib	PP Lib
MUX-I	Basic Lib	PP Lib
MUX-MI	User_Lib	PP Lib
DATE	Basic Lib	

TIME	Basic Lib	
SW	Basic Lib	PP Lib
ABS	Basic Lib	
REPORT	User_Lib	
FILT-1P	User_Lib	PP Lib
CON-PU1		PP Lib
PB-R	Basic Lib	
SR-OA	Basic Lib	
MAJ-R	User_Lib	
MUXGR-MI	User_Lib	
PB-S	Basic Lib	
XOR	Basic Lib	
SQRT	Basic Lib	
ADD-MR	Basic Lib	PP Lib
ADD-MR1	Basic Lib	
FIFO		PP Lib
DEMUX-MI		PP Lib
DER		PP Lib
FILT-2P		PP Lib
REG-G	User_Lib	PP Lib
FUNG-1V	User_Lib	PP Lib
MAX	Basic Lib	PP Lib
MIN	Basic Lib	PP Lib
PI	ControlSimpleLib	PP Lib
PDP	User_Lib	User_Lib
RAMP	ControlSimpleLib	PP Lib
RAMP-S1	User_Lib	User_Lib
TIMER	User_Lib	PP Lib
MANSTN		PP Lib
MOTCON		PP Lib
VALVECON		PP Lib
PIDCON		PP Lib
SEQ	only SFC	PP Lib, SFC
STEP	only SFC	PP Lib, SFC
DIV-MR	Basic Lib	
TEXT	User_Lib	
RATIOSTN		PP Lib
EXP	Basic Lib	
LN	Basic Lib	
BGET	Basic Lib	
BSET	User_Lib	
IIL	User_Lib	
ILI	User_Lib	
DRTRA (APC)	User_Lib	

Developed by: Kamil Herman, Slovakia
contact: kamil.herman@herman-automation.com

ABB AC450 and 800xA Specialist Providing a range of Control System design, programming, migrating and commissioning services. We are ready to deliver the complete migrated control software in status „Ready for FAT“ or „Ready for Commissioning“.