SECTION 25 5256

Building Automation Systems (BAS) Programming

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

LANL MASTER SPECIFICATION SECTION

Word file at [https://engstandards.lanl.gov/specs.shtml](https://engstandards.lanl.gov/specs.shtml#25)

|  |
| --- |
| Rev. 0 Summary of Changes  Initial issue. |

**When the project includes a BAS, this template material is to be included in the Project Specification, either as this standalone section or by incorporation of the material herein into 25 5000, *Integrated Automated Facility Controls*.**

This template must be edited for each project. In doing so, Specifier must add job-specific requirements. Brackets are used in the text to indicate designer choices or locations where text must be supplied by the designer.  Once the choice is made or text supplied, remove the brackets.  This section must also be edited to delete requirements for processes, items, or designs that are not included in the project -- and specifier’s notes such as these.

To seek a variance from requirements in this Section that are applicable, contact the Engineering Standards Manual Chapter 8[POC](http://engstandards.lanl.gov/POCs.shtml#ic). Please contact POC with suggestions for improvement as well.  
  
When assembling a specification package, include applicable sections from all Divisions, especially Division 1, General requirements.  
  
Section developed for ML-4 projects.  For ML-1, 2, and 3 applications, additional requirements and independent reviews should be added if increased confidence in procurement or execution is desired; see ESM Chapter 1 Section Z10 Att F Specifications and Quality sections.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. GENERAL
   1. Section Includes
      1. Program naming conventions.
      2. Program comment and revision tracking conventions.
      3. Program ties to network trees (third-party controllers)
      4. Base program generation.
      5. Use of LANL standard programs.
      6. Restricted program features.
      7. Preferred programing techniques.
   2. LANL-PERFORMED WORK
      1. None
   3. SUBMITTALS
      1. Programs shall be submitted for review as required in Section 25 5000, *Integrated Automated Facility Controls.*
   4. Coordination
      1. Verify equipment and component and equipment designations with LANL through the Subcontract Technical Representative (STR) if they are not specifically designated on design documents or clearly stated in this Section. Do not arbitrarily assign designations.
   5. Applicability
      1. BAS that require programs to implement the required sequence of operations specific to the system in which they are installed. This Section does not apply to factory provided drivers, operating systems, development systems, or other similar programs that cannot be modified by the installer.
      2. This Section is written for Automated Logic Controls Eikon programs specifically but should be applied to all BAS programming from any manufacturer (e.g., Trane, Alerton, JCI, etc.) to the extent they are allowed, and it is applicable.
   6. SCOPE LIMITATIONS
      1. This Section shall not be applied to ML-1, 2, or 3 programs without express written permission from the LANL ESM Chapter 8 POC.
2. PRODUCTS (programing requirements)
   1. programming naming CONVENTIONS
      1. Provide every program with a unique name consisting of:
         1. The LANL TA number in the format TA##, where ## is the TA number using two digits. When no single TA is associated with the program, use 99 as the TA number.
         2. A hyphen ( “-“ ) to separate the TA from the building number
         3. The LANL building or structure number in the format ####, where #### is the building or structure number using 4 digits. Right-justify and pad numbers with leading “0s” as needed to form a four-digit number (e.g., building 3 would be displayed as “0003”). Use MULT in place of the number when the program applies to multiple buildings or structures; INFR when infrastructure such as utilities.
         4. A space ( “ ” ) to separate the building number from the equipment tag(s).
         5. The tag name of the equipment served including the identification number(s). For example: “AHU-1”, “HVA-2, 5, 7” or “RUA-1 to 8”. If the program applies to multiple equipment tag names, list all; for example: “FE-2 and OAU-1.” On rare occasions, where no equipment tag name is available, the optional short function description (item 7 below) becomes mandatory.
         6. A space ( “ ” ) to separate the equipment tag from the optional description.
         7. Optionally, a short function description if needed to clarify the purpose of the program or special characteristic of the program. For example: “with SA temp reset,” “equipment shutdown,” or “with equipment room temp sensor.”
      2. Examples of acceptable program names are:
         1. TA02-0235 AHU-1
         2. TA99-MULT Network Monitoring
         3. TA03-0001 VAV1 to 34, 102 to 105 with reheat coils.
      3. Display the program name in bold, large, white lettering at the upper left-hand corner of the program (figures herein use reversed coloration for clarity):

Graphical user interface, text, application

Description automatically generated with medium confidence

Figure 1 – Example Program Title

* + 1. When the program is stored, the program name shall be used as the file name except all characters shall be lower case and all spaces shall be replaced with underscores to maintain compatibility with Eikon. The current program revision shall be added to the program file name using the format “\_rev\_a” or “\_rev\_100.” Drop the decimal point from the numbered revisions to avoid confusion with the file type identifier (e.g., Rev 1.01 would be “\_rev\_101”, Rev 16.23 would be “\_rev\_1623”). Programs shall use the “.equipment” as the file type. For example: “TA02-0235\_AHU-1” at revision level 1.03 shall be stored to file “ta02-0235\_ahu-1\_rev\_103.equipment.” Optionally, the program file name may be displayed below the program title in the program using standard-size, white font.
  1. Program comment and revision tracking conventions.
     1. Programs shall be adequately documented using comments and formatting so those unfamiliar with purpose and operation of the program can understand it. The following are the minimum requirements:
        1. Subroutines: In graphical programming, subroutines are sections of the program addressing a specific task. Each subroutine is enclosed in a brown rectangle with the title of the subroutine, in brown standard font, located in the top left of the rectangle. The title is proceeded and followed by a space and placed on top of the rectangle to assure maximum readability.   
             
           Exception: For critical subroutines for ML-3 applications, fume hood control, laboratory pressure controls, and other life safety related controls, the color of the subroutine rectangle and title shall be red to indicate caution must be taken when modifying or overriding items contained within.   
             
           Provide a brief text description of the subroutine within the bounding rectangle if the function of the subroutine is complex or hard to understand from the logic.   
             
           Subroutines may be nested.
        2. Microblock Labels: When the parameters of the block are expected to be changed or the function of the block is unclear, provide text labels for to clarify the function. Label all parameter microblocks that contain set points.
        3. Property Pages: Use of property pages is optional. Property page items may be turned off for all programs manually or by use of the Eikon built-in tool “Remove all Property Page Text”. If property pages are left active in the program, they shall be properly formatted and logically organized to display correctly.

Diagram

Description automatically generated with medium confidence

Figure 2 – Example Comments

* + - 1. Inputs and Outputs: Input microblocks shall always be located on the far left of the program screen. Output microblocks shall always be located on the far right of the program screen. The Display Name field of the input and output microblocks shall match the noun name (e.g., “SA Temp”, “Cooling 1”, “Supply Fan Speed”) given on the P&ID or the points list provided as part of the Drawings. The Description field of input and output blocks shall be used to identify any P&ID tag names (e.g., “YA‑1001”, “TT-101”, “FCV-202”) and may optionally contain additional text to clarify the function of the point.
      2. Network Points: Microblocks used to connect external network points shall include comments indicating the source and destination of the data on both ends of the network connection. For instance, a fan interlock point sourced on controller “A” from a BACnet parameter and received on controller “B” by a network-read shall label the source parameter “Fan Interlock to Controller B,” and the network-read labeled “Fan Interlock from Controller A.”
      3. Order of Subroutines: The order of the subroutines in the program, from top to bottom, should implement the order of the sequence of operations from the design documents.
      4. Crossing of Connection Lines: Do not cross microblock connection lines unless they are connected. Rearrange microblocks to avoid crossing connection lines or use labels to cross unconnected lines.
    1. Programs shall include a revision block at the bottom of the program listing to track all changes to the program. It shall be contained in a subroutine identification block with the title “Revisions.” Revision identification shall use letters (e.g., Rev A) for revisions made during development and initial testing. Once a program is released for final commissioning it shall use the number format XX.YY, where XX is the major revision number and the YY is the minor revision number (e.g., Rev 1.03). See ESM [Chapter 21](https://engstandards.lanl.gov/ESM_Chapters.shtml#esm21) (Section SOFT-GEN) for a more complete expectations for revision numbering and the difference between major and minor program changes. Revision entries shall follow the following requirements:
       1. Each revision entry shall contain the following information:
          1. The revision identification number or letter (e.g., Rev 2.05) followed by a colon ( “:” ).
          2. The date of the revision in the format MM/DD/YY (e.g., 07/29/21).
          3. The name or initials of the person making the changes (e.g., A. Hayward, or ADH).
          4. The company of the person making the changes (e.g., LANL or ICSI) followed by an en dash ( “– “ ) or hyphen — and multiple ones for multi-line descriptions.
          5. A brief description of the change or changes (e.g., Added supply air reset, or Corrected compressor sequencing). Use multiple indented lines if the description will not fin on one line (see figure below for example).

Text

Description automatically generated with medium confidenceFigure 3 - Example Revision Block

* + - 1. If a legacy program does not include a revision block and revisions are needed, add a revision block and indicate any known information about the origin of the original program.
      2. If a program is being repurposed to a new location, include the original program name in the new revision block entry.
  1. Program ties to network trees (third-party controllers)
     1. When third-party controllers are integrated into the BAS, the associated program(s) must reside on controllers that accept Eikon programming. To reduce confusion, do the following:
        1. The third-party BACnet controller must be entered into the WebCTRL network tree as a BACNET controller and shown at the actual physical connection point it is installed on the network. This controller will not accept an Eikon program, but this network tree entry does provide a way to document the physical location of the device on the network as well as its BACnet Device Instance and MAC address for troubleshooting purposes.
        2. Comments should be added to the Eikon programs to clarify the destination or source controller of network-connected points.
  2. Base program generation
     1. When generating new code, use Equipment Builder software to generate the base code. Modify this code as needed to meet the specific sequence of operations.
     2. When Equipment Builder does not provide appropriate equipment selections, generate new custom code, but follow the Equipment Builder format and naming conventions to the maximum extent possible.
     3. The reuse of existing, proven code is encouraged. When possible, use existing LANL BAS programs and repurpose for new applications. Update program names and comments as needed for the new application.
  3. Use of LANL standard programs
     1. LANL provides several standard programs that should be reused in new applications. These programs are provided with an application guide that assists the user with how and when to use these programs. LANL programs are provided in a generic form and must be edited for final use by the programmer. Many LANL programs are supplied with example View Builder files that should be used in the graphic interface design. When a LANL program provides the same function as an Equipment Builder program, the LANL program shall be used in lieu of the Equipment Builder code.
     2. Check with LANL’s Engineering Standards I&C POC and/or ESM Chapter 8 webpage for the most up-to-date list of LANL provided programs. The following is a short list of some of the LANL programs available:
        1. Lead-Standby and Lead-Lag Switching of redundant systems,
        2. Fume hood lab controls,
        3. ABB Variable Frequency Drive (VFD) BACnet Interface,
        4. Outside air makeup supply systems (in development).
  4. Restricted program features
     1. Although the Eikon programming language supports the following functions and features, due to readability and understandability the use of the following should be limited. When used, provide adequate commenting to make the use clear.
        1. Microblock Macros (Custom Microblocks) – The use of this feature limits the visibility of the underlying code when viewed though the WebCTRL logic page. Name the custom microblock with a logical name and use appropriate input and output point names that convey their use clearly. Add text comments near the Custom Microblock that explain the function of the block.
        2. OCL Microblocks – This block uses the Operator Control Language (OCL) to create special functions not typically available using microblocks. This language is not well understood by most Eikon programmers making it hard to understand, debug, and modify. Use of this type of microblock should be limited to applications that cannot be implemented in with standard microblock logic.
     2. Deadwood Removal – The Equipment Builder program, due to its generic design nature, inserts one or more subroutines at the bottom of the generated program providing constant links and glue logic. The use of these subroutines reduces the readability of the main program logic and in many cases inserts unused code (deadwood). These subroutines shall be removed, and the main program patched to remove dependency on these sections.

Graphical user interface, application

Description automatically generated

Figure 4 - Example of Deadwood

* 1. Preferred programing techniques
     1. Use of wire lock microblocks – Use wire lock microblocks (analog and digital) to provide override points in the program in locations where operators or testing personnel may be expected to override normal program operations. Wire locks are preferred over traditional locking mechanisms because they can be set with time limits that automatically release the lock at a specified time or duration. The use of duration-based overrides can reduce the possibility of a testing or temporary lock being inadvertently left in the locked condition.

Diagram

Description automatically generated

Figure 5 - Locking Options

* + 1. Use of BACnet PID microblocks – Use the BACnetPID microblock whenever a PID loop controller is needed in place of the legacy PID-Direct Acting or PID-Reverse Acting microblocks. The BACnet PID block provides significantly better tuning capability and the ability to reverse the direction of the operations without having to modify the programming.

A picture containing diagram

Description automatically generated

Figure 6 - PID Microblocks

* + 1. Use of BACnet parameter microblocks – When using parameters in a program, use the BACnet version of the parameter microblocks (analog, digital, and multi-state) in place of the legacy Eikon parameter microblocks. This allows full compatibility with other BACnet devices, the ability to share the parameter values over the network and additional features only available in the BACnet version of the microblock.

Graphical user interface, text, application

Description automatically generated

Figure 7 - Parameter Microblocks

* + 1. When transferring data between BACnet controllers, the preference is to use Network Read microblocks over Network Write microblock. This can reduce the possibility of accidently writing data to the wrong controller or point due to a network address error or a change in controller addressing. This technique is also easier to troubleshoot.
    2. Hardwired safety shutdowns, as described in Section 25 5911, shall:
       1. Automatically clear when the hardwired shutdown condition is cleared and allow the system to return to normal operations with no need to access the WebCTRL user interface (i.e., no software latches that must be reset to restore operations).
       2. Be mirrored in the software to match hardwired shutdown action. For instance, if the hardwired shutdown forces the chilled water valve fully open, then the software shall also command the chilled water valve fully open while in shutdown mode.

1. EXECUTION
2. DEVELOPMENT
3. General Qualifications of Programmer: Personnel performing BAS programming shall:
   1. Have (a) at least one factory provided class on Eikon programming and (b) at least one year of Eikon programming experience or shall be under the direct supervision of person who such experience.
   2. Failing this, provide proof of equivalent programming experience and be approved by LANL BAS Lead SME.
4. The requirements of this Section shall be implemented in the programs prior to start of final commissioning of the BAS.
   1. acceptance
      1. Failure to meet the requirement of this Section shall result in rejection of the provided program(s). It is the responsibility of the program provider to make corrections to bring the program into compliance with this Section at no extra cost to LANL.
      2. Initial program review and approval shall be done by [a LANL I&C Design engineer] [and] [or] [a LANL BAS Team member]. Rejection of program(s) may be appealed to the project’s BAS Responsible Design Engineer or to the LANL I&C Lead SME for reconsideration.
   2. ARchives
      1. Integration into the main LANL BAS server:
         1. All Eikon source programs (.equipment files) shall be provided to the BAS team when the new control system is integrated into the main LANL BAS server or the LANL BAS Development server as part of the database update. The LANL BAS team shall assure that reliable archives of the programs are maintained as part of the server backup process.
      2. Integration into LANL standalone servers:
         1. All Eikon source programs (.equipment files) shall be provided as part of the database update to the standalone server.
         2. The owner of the standalone server is responsible for maintaining reliable archives of the program(s) located on their server.
      3. Old versions of programs shall be maintained, either on the server or as part of a separate back up. When modifying programs, do not write new program files over the original files. Use new revision numbers with the file name to assure unique names for each program version.

END OF SECTION

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Do not delete the following reference information.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

THE FOLLOWING STATEMENT IS FOR LANL USE ONLY

This project Specification section is based on LANL Master Specification Section 25 5256, Rev. 0, dated April 20, 2023.