

Presentation

Concept is a software configuration and application programming tool for the Momentum automation platform. It is a Windows-based software that can be run on a standard personal computer. The configuration task can be carried out online (with the PC connected to the Momentum CPU) or offline (PC only). Concept supports the configuration by recommending only permissible combinations, thereby preventing misconfiguration. During online operation, the configured hardware is checked immediately for validity, and illegal statements are rejected.

When the connection between programming unit (PC) and Momentum CPU is established, the configured values (e.g., from the variables editor) are checked and compared with actual hardware resources. If a mismatch is detected, an error message is issued.

Concept editors support five IEC programming languages:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Sequential function chart (SFC)
- Instruction list (IL)
- Structured text (ST)

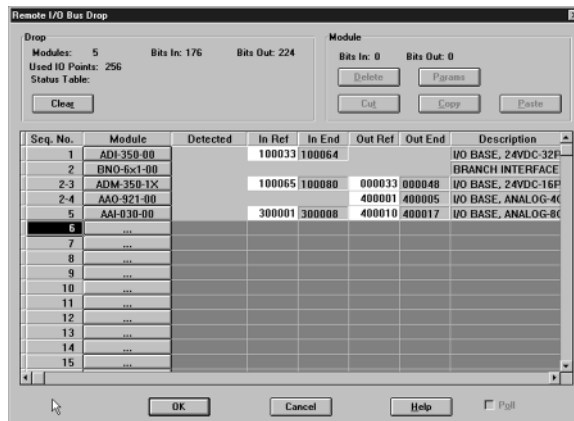
as well as Modsoft-compatible ladder logic (LL984). IEC 1131-3 compliant data types are also available. With the data type editor, custom data types can be converted to and from the IEC data types.

The basic elements of the FBD programming language are functions and function blocks that can be combined to create a logical unit. The same basic elements are used in the LD programming language; additionally, LD provides contact and coil elements. The SFC programming language uses basic step, transition, connection, branch, join and jump elements. The IL and ST text programming languages use instructions, expressions, and key words. The LL984 programming language uses an instruction set and contact and coil elements.

You can write your control program in logical segments. A segment can be a functional unit, such as conveyor belt control. Only one programming language is used within a given segment. You build the control program, which the automation device uses to control the process, by combining segments within one program. Within the program, IEC segments (written in FBD, LD, SFC, IL and ST) can be merged. The LL984 segments are always processed as a block by the IEC segments. Concept's sophisticated user interface uses windows and menus for easy navigation. Commands can be selected and executed quickly and easily using a mouse. Context-sensitive help is available at each editing step.

PLC hardware configuration

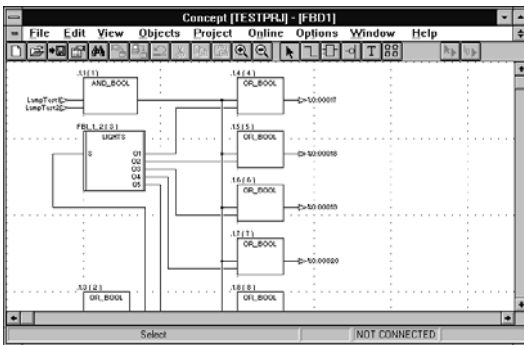
Variables for linking basic objects within one section are not required by the graphic programming languages (FBD, LD, SFC and LL984) since these links are created by connections. These connections are managed by the system, which eliminates any configuration effort. Other variables, such as variables for data transfers between different sections, are configured with the variables editor. With the data type editor, custom data types can be derived from existing data types.



Concept provides an editor for each programming language. These editors contain custom menus and tool bars. You can select the editor to be used as you create each program segment.

In addition to the language editors, Concept provides a data type editor, a variables editor and a reference data editor.

Function block diagram (FBD)



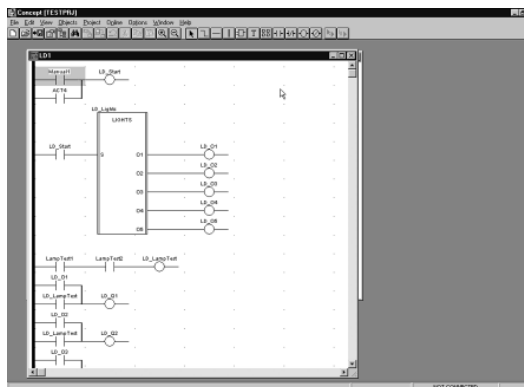
With the IEC 1131-3 function block diagram language, you can combine elementary functions, elementary function blocks (EFBs) and derived function blocks (all three of which are known as FFBs) with variables in an FBD. FFBs and variables can be commented. Text can be freely placed within the graphic. Many FFBs offer an option for input extensions.

Concept provides various block libraries with predefined EFBs for programming an FBD. EFBs are grouped in the libraries according to application types to facilitate the search.

In the FBD editor, you can display, modify and load initial values; current values can be displayed. The CLC and CLC_PRO libraries allow you to display animated diagrams of the FFBs and a graph of the current values.

For custom function blocks (DFBs), the Concept-DFB editor is used. In this editor, you can create your own function blocks from EFBs or existing DFBs. DFBs created in the FBD editor can be recalled in the LD, IL and ST editors, and DFBs created in the LD, IL and ST editors can be used in the FBD editor.

Ladder diagram (LD)



With the IEC 1131-3 ladder diagram language, you can build an LD program with elementary functions, function blocks and derived function blocks (all of which are known as FFBs), along with contacts, coils and variables. FFBs, contacts, coils and variables can be commented. Text can be placed freely within the graphics. Many FFBs offer an option for input extensions.

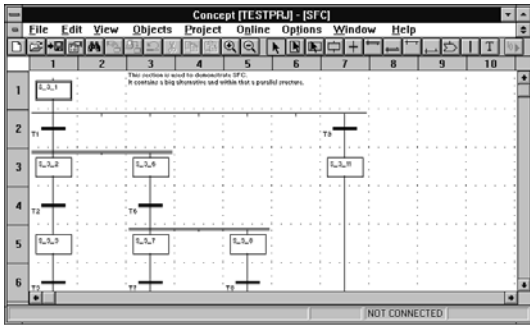
The structure of an LD segment corresponds to that of a current path for relay circuits. On its left side is a left bus bar, which corresponds to the phase (L conductor) of a current path. As with a current path, only the LD objects (contacts, coils) connected to a power supply (i.e., connected to the left bus bar) are processed in LD programming. The right bus bar, which corresponds to the neutral conductor, is not visible. However, all coils and FFB outputs are internally connected to it in order to create a current flow.

The same EFB block libraries available for the FBD editor can be used in the LD editor to program a ladder diagram.

In the LD editor, initial values can be displayed, modified and loaded; current values can be displayed. For the EFBs in libraries CLC and CLC_PRO, animated diagrams of the FFBs and a graph of the current values can be displayed.

For custom function blocks (DFBs), the Concept-DFB editor is used. With this editor, you can create your own function blocks from EFBs or existing DFBs. DFBs created in the LD editor can be recalled in the FBD, IL and ST editors, and DFBs created in the FBD, IL and ST editors can be used in the LD editor.

Sequential function chart (SFC)



With the IEC 1131-3 sequential function chart (SFC) language, you can define a series of SFC objects that comprise a control sequence. Steps, transitions and jumps in the sequence can be commented. You can place text freely within graphics. You can assign any number of actions to every step. A series of monitoring functions—e.g., maximum and minimum monitoring time—can be integrated into each step's characteristics. The actions can be assigned an attribute symbol (as required by IEC) to control the action's performance after it has been activated—e.g., a variable can be set to remain active after exiting.

Instruction list (IL)

With the IEC 1131-3 IL language, you can call entire functions and function blocks conditionally or unconditionally, execute assignments and make conditional and unconditional jumps within a program segment.

IL is a text-based language, and standard Windows word processing tools can be used to generate code. The IL editor also provides several word processing commands. Keywords, separators and comments are spell-checked automatically as they are entered. Errors are highlighted in color.

For custom function blocks (DFBs), the Concept-DFB editor is used. In this editor, you can create your own function blocks from EFBs or existing DFBs. DFBs created in the IL editor can be recalled in the ST, LD and FBD editors, and DFBs created in the ST, LD and FBD editors can be used in the IL editor.

Structured text (ST)

```

ST1
===== Structured Text Start =====
VAR
TIMER : TON;
END_VAR

TIMER(IN := NOT pulse,
PT := T#1s); (* Blink timer *)
pulse := TIMER.Q;

(* Count every pulse *)
IF pulse = 1 THEN
count := count + 1;
END_IF;
(* Animate lights according to count *)
CASE count OF
1: out1 := TRUE;
2: out2 := TRUE;
3: out3 := TRUE;
END_CASE

IL1
===== Instruction List Start =====
VAR
RUN_TIMER : TON; (* Blink timer *)
END_VAR

(* Default for the marker *)
LD
run_light1
run_light
ST
run_pulse

(* Create a 1.0 Hz. pulse *)
LD
run_pulse
STN RUN_TIMER.IN
CAL RUN_TIMER(PT := T#1s)
LD
RUN_TIMER.ET
ST
animateLine
LD
RUN_TIMER.Q
ST
run_pulse
JMPEN end (* No pulse y
    
```

With the IEC 1131-3 ST language, you can call function blocks, execute functions and assignments and conditionally execute and repeat instructions. The ST programming environment is similar to Pascal. It is a text-based language, and Windows word processing functions can be used to enter code. The ST editor itself also provides several word processing commands. Keywords, separators, and comments are spell-checked automatically as they are entered. Errors are highlighted in color.

Custom function blocks (DFBs) created with the ST editor can be called in the IL, LD and FBD editors; DFBs created in the IL, LD and FBD editors can be used in the ST editor.

Data type editor

The data type editor defines new derived data types. Any elementary data types and derived data types already existing in a project can be used for defining new data types. With derived data types, various block parameters can be transferred as one set. Within the program, this set is divided again into single parameters, processed, then output as either a parameter set or individual parameters. Derived data types are defined in text format, and standard Windows word processing tools can be used. The data type editor also provides several word processing commands.

Variables editor

The variables editor contains input options for:

- The variable type (located variable, unlocated variable, constant)
- The symbolic name
- The data type
- Direct address (explicit, if desired)
- Comments
- Identification as human-machine interface (HMI) variable for data exchange

Reference data editor

In online mode, the reference data editor displays, forces and controls variables. The editor contains the following options:

- Default values for the variable
- Status display for the variable
- Various format definitions
- The ability to isolate the variable from the process

Libraries

IEC Library

The IEC library contains the EFBs defined in IEC 1131-3 (calculations, counters, timers, etc).

Extended Library

The extended library contains useful supplements to various libraries. It provides EFBs for mean value creation, maximum value selection, negation, triggering, converting, building a traverse with interpolation of the first order, edge detection and determination of the neutral range for process variables.

System Library

The system library contains EFBs in support of system functions. It provides EFBs for cycle time detection, utilization of various system clocks, control of SFC sections and system status display.

CLC and CLC_PRO Library

The CLC library is used for defining process-specific control loops. It contains control, differentiation, integration and polygon graph EFBs. The CLC_PRO library contains the same EFBs as the CLC library along with data structures.

Communication Library

The communication libraries of built-in function blocks provide easy integration of programs which allow communication between PLCs or HMI devices from within the PLC's application program. Like other function blocks, these EFBs can be used in all languages to share data, or provide data to the HMI device for display to the operator.

Diagnostics Library

The diagnostics library is used for troubleshooting the control program. It contains EFBs for action, reaction, interlocking, and process prerequisite diagnostics, along with signal monitoring.

LIB984 Library

The LIB984 library provides common function blocks used in both the 984 ladder logic editor and the IEC languages. This allows for easy transition of portions of application code from the 984LL environment to the IEC environment.

Fuzzy Logic Library

The fuzzy library contains EFBs for fuzzy logic.

Analog I/O Library

The ANA_IO library is used to process analog values.

The ProWORX programming software is a full-featured, Modicon PLC programming software that is compatible with any Windows platform - 3.1/95/98/NT. A few of the new ProWORX features follow:

Windows environment

The familiar Windows-based programming environment means you spend less time learning how to do things, and more time being productive. ProWORX uses familiar Windows features like user-defined screens, drag-and-drop, cut and paste, search, and global replace.

Intuitive Register Editor

A powerful analysis tool, the Data Watch Window shows you information from your plant in real-time, or logs it to disk for in-depth historical analysis later on. Easily get the data you need to make informed, effective production decisions. View and edit data in full page display, see trends and track data points against time in a spreadsheet, and monitor any combinations of discretes and analogs.

I/O drawing generator

Save hours of painstaking effort with ProWORX NxT's I/O Drawing Generator, which automatically creates wiring diagrams for the I/O cards defined in the Traffic Cop. Generate necessary drawings all at once or just one card at a time – simply select an address the I/O card uses with the Network Editor, then click the drawing button on the Hardware Back Referencing panel. NxT displays the diagram, and if desired, saves it as an AUTOCAD-compatible .DXF file or prints it

Network editor

With the Network Editor, ProWORX NxT reduces development time by using the same commands and instructions for every controller. Simply cut, copy, and paste networks from one platform to any other.

Real-time network status

Find the controller you need fast and simplify network diagnostics with ProWORX NxT's powerful Network Scan feature. Network Scan searches your Modbus or Modbus Plus networks, then identifies and graphically displays each device found and shows its status.

Advanced I/O management

Ensure that the I/O card you are configuring in the software matches the one on your plant floor with Pro WORX NxT's graphical Traffic Cop. It displays I/O cards on your screen the same way they look in real life, eliminating all confusion. To place a card, just select it from the convenient drop down menu and then drag it into the controller slot you want. To save even more time, the Traffic Cop automatically associates the card's I/O points with with a block of free addresses in your controller. Once configured, manage your I/O with NxT's complete documentation tools, with references for each head, drop, rack, slot and address. And the Traffic Cop's graphical display shows you at a glance that your I/O is healthy.

Momentum automation platform

Programming softwares

Concept

References

Concept softwares

Description	License type	References	Weight kg
Concept packages			
Concept S Version 2.5	single-user license	372 SPU 471 01 V25	–
Concept M Version 2.5	single-user license	372 SPU 472 01 V25	–
Concept XL Version 2.5	single-user license	372 SPU 474 01 V25	–
	three-user license	372 SPU 474 11 V25	–
	10-user license	372 SPU 474 21 V25	–
	network license	372 SPU 474 31 V25	–
Concept EFB Toolkit Version 2.5	–	372 SPU 470 01 V25	–

Concept upgrades

Description	License type	References	Weight kg
Concept XL V. 2.2 to Concept XL V.2.5	single-user license	372 ESS 474 01	–
	three-user license	372 ESS 474 03	–
	10-user license	372 ESS 474 10	–
	network license	372 ESS 474 00	–
Concept S V. 2.2 to Concept S V. 2.5	single-user license	372 ESS 471 01	–
Concept M V. 2.2 to Concept M V. 2.2	single-user license	372 ESS 472 01	–
Modsoft V x.xx to Concept XL V.2.5	depends on number of users	372 ESS 485 01	–
Concept EFB Toolkit V x.x to V 2.5	–	372 ESS 471 01	–

Documentation

Description	Number of volumes	References (1)	Weight kg
Concept Installation Instructions	1	840 USE 492 0●	–
Concept User Manual	3	840 USE 493 0●	–
Concept IEC Block Library	13	840 USE 494 0●	–
Concept 984 LL Block Library	2	840 USE 496 0●	–
Concept EFB Tool User Manual	1	840 USE 495 01	–

(1) ● = 0 in this position indicates english language, 1 indicates french language, 2 indicates german language and 3 indicates spanish language.

Momentum automation platform

Programming softwares

ProWORX

ProWORX softwares			
Description	License type	References (1)	Weight kg
ProWORX packages			
ProWORX NxT Online	single-user license	372 SPU 681 01 NONL	—
ProWORX NxT Offline/ Online	single-user license	372 SPU 680 01 NDEV	—
	3-user license	372 SPU 680 01 NSTH	—
	10-user license	372 SPU 680 01 NSTE	—
	20-user license	372 SPU 680 01 NSTW	—
ProWORX NxT Lite Offline/Online	single-user license	372 SPU 610 01 NLDV	—
	3-user license	372 SPU 610 01 NLTH	—
	10-user license	372 SPU 610 01 NLTE	—
	20-user license	372 SPU 610 01 NLTW	—
ProWORX Upgrades Modsoft Upgrade to ProWORX NDEV	single-user license	372 SPU 684 01 NXUP	—
	3-user license	372 SPU 684 01 MSTH	—
	10-user license	372 SPU 684 01 MSTE	—
	20-user license	372 SPU 684 01 MSTW	—
ProWORX Plus Upgrade to NxT NDEV	single-user license	372 SPU 684 01 NXPW	—
	3-user license	372 SPU 684 01 NPTH	—
	10-user license	372 SPU 684 01 NPTE	—
	20-user license	372 SPU 684 01 NPTW	—
ProWORX NxT Online/ Offline Development	single-user license	372 SPU 610 01 DEV	—

ProWORX Client/Server softwares			
Description	User	References	Weight kg
ProWORX packages			
ProWORX NxT 32	Server	372 SPU 780 01 PSEV	—
		372 SPU 780 01 PSSV	—
	Offline/Online Client	372 SPU 780 01 PDEV	—
	Online Client	372 SPU 781 01 PONL	—
ProWORX NxT 32 Lite	Offline/Online Client	372 SPU 710 01 PLDV	—
Legacy Product Upgrade to NX32	Client	372 SPU 784 01 LPUP	—
	Multiuser Incremental Addition	372 SPU 780 01 SEAT	—
ProWORX 32	3 Multi-user Client License	372 SPU 780 01 PSTH	—
	10 Multi-user Client License	372 SPU 780 01 PSTE	—
ProWORX 32 Lite	3 Multi-user Client License	372 SPU 710 01 PLTH	—
	10 Multi-user Client License	372 SPU 710 01 PLTE	—
Legacy Product Upgrade to NX32	3 Multi-user Client License	372 SPU 784 01 LPTH	—
	10 Multi-user Client License	372 SPU 784 01 LPTE	—

Momentum automation platform

Programming softwares

ProWORX

ProWORX softwares (continued)

Documentation			
Description	Language	Reference	Weight kg
ProWORX NxT Programming Software User Manual	–	372 SPU 680 01 NMAN	–
ProWORX 32 User Manual	English	372 SPU 780 01 EMAN	–
	French	372 SPU 780 01 FMAN	–
	German	372 SPU 780 01 DMAN	–
	Spanish	372 SPU 780 01 SMAN	–