iCC 1994

1st international CAN Conference

in Mainz (Germany)

Sponsored by

Allen Bradley
National Semiconductor
Philips Semiconductors

Organized by

CAN in Automation (CiA)
international users and manufacturers group
Am Weichselgarten 26
D-91058 Erlangen
Phone +49-9131-69086-0
Fax +49-9131-69086-79
Email:headquarters@can-cia.de
URL: http://www.can-cia.de

Modelling Simatic S5 with CMS

Sönke Hansen
JANZ Computer AG
Im Dörener Feld 8
D-33100 Paderborn

The Simatic S5 PLC

The Simatic S5 is a programmable logic controller (PLC). It is widely used in automation for controlling industrial processes. A PLC typically executes an application program cyclically, reading all inputs at the beginning of a cycle, changing the values of operands in memory during a cycle, and writing to the output devices at the end of a cycle.

Communication processors are used to attach a S5 serially or via a bus to other intelligent partners. Typically a point-to-point protocol connecting two communication partners is implemented. A partner may either send data on his own initiative (SEND job) or request data from the partner (FETCH job). The parameters of communication jobs specify the type of operand (input, output, timer, counter, data, etc.) to be transferred, its location in memory (e.g., DB and DW), and its length in number of bytes or words depending on the operand type.

Often applications regard a S5 including the I/Os it controls as a single unit which mainly operates autonomously. The application may then be interested only in certain state changes in the PLC. This calls for an interface to a PLC which allows applications to specify events about which it wishes to be notified. The CAN fieldbus is well suited for event driven communication. In the following an interface is described which connects a S5 to a CAN bus. This interface conforms to the CAN Application Layer CAL. The emphasis is on the choice of CMS objects describing the interface. The interface is designed to make it easy for the network builder to integrate one or several PLCs into his CAL network.

CAN Application Layer

CAL gives manufacturers of modules and builders of CAN networks a flexible setting for doing their work without restricting the freedom of others. The manufacturer of a module defines its functionality in terms of CMS objects leaving the task of assigning CAN identifiers to the network builder where it belongs. The network builder assigns network specific parameters for CMS objects (identifiers, inhibit times) using DBT services. LMT services transfer a module from the manufacturers world into the network builders world (setting of NMT address and baud rate). Knowledge about modules and their operation is confined to a special service element, NMT. CMS, on the other hand, knows about other modules only indirectly through variables, domains, and events. In a CAL network with all modules operational the application consists of the modules acting as clients or servers of the CMS objects defined on them. CAL is an efficient application layer protocol. Compared with a pure layer-2 protocol no overhead is incurred when using CAL in a fully operational network except for the error control services (guarding protocol) if used.

It is the task of the manufacturer of a module to specify its functionality and communicate this on a data sheet to the CAL network builder. A specification may state that a module implements the full CAL slave capabilities, i.e., LMT class 2, NMT node class 4, and DBT class 2. Such a module works well in a CAL network of class at least 1. The main part of the specification of a CAL module then consists of the description of the CMS objects which it supports including their semantics. The network builder also needs to know which configuration data can be downloaded to a module with NMT configuration services. These services can provide a means to influence the actual definition of CMS objects.

CMS Model of Simatic S5

In the following the CMS objects of a simplified S5-CAN interface module are described. A S5 PLC maintains process images consisting of contiguous operand ranges with operands of a given type (inputs, outputs, data). It should be kept in mind that the number of I/Os controlled by a single S5 is typically quite large, e.g., several hundred. Here the PLC is viewed as a large array of inputs which can be read and on which events can be defined.

A CMS object named #S5Image (free format name with blanks padded to the left) is defined with the Define Domain service as a basic domain with user type server. The definition takes place in the node state preparing after the appropriate configuration data are downloaded from the NMT Master (Configuration Download). The data downloaded consist of the parameters necessary to specify a S5 image: data type (e.g., E for input), the data block number DB, the address of the data DW, and the data length. The corresponding S5 process image is cyclically updated in the module through FETCH requests. A client can read the part of the process image so defined with the Domain Upload service. Hence the client of #S5Image can read the current process image in the S5 if desired.

A CMS object named #Poll is defined as a write-only variable of user type client with data type boolean. When there is a server of this object writing the value true then the module starts a download of #S5Image to the client of the domain. The purpose of variable #Poll is to force synchronized cyclical updates (polling).

Depending on the NMT configuration data a list of Stored Events with user type server is defined. The configuration data needed for defining a single event consists of: The CMS name, say #S5Event1, the subdomain of #S5Image to which the event refers (relative start address, length L), and an event mask of data type Unsigned(8 * L). These parameters define a subarray, the value of the event, of the array making up the process image. The data type of #S5Event1 is Unsigned(8 * L). The value of L may not exceed 8. The cyclical updates of the process images from the S5 result in services Store Event for #S5Event1. The service parameter immediate-notify is true precisely when the value of a non-masked bit has changed in the update. Thus clients of #S5Event1 are notified about a change in non-masked bits in an subarray of up to 8 bytes length once the change occurs. If the subdomain in which events may occur cannot be confined to a subarray of length at most 8 then more than one CMS object of type Stored Event is needed. If this would consume too many identifiers an alternative should be considered: using Uncontrolled Events multiplexing is feasible.

Clearly these CMS objects give only a partial picture of a S5 PLC. However they indicate a path for a manufacturer of S5-CAN modules to follow when specifying the capabilities of a module to a network builder. They also show that the manufacturer need not involve himself into details of a specific application when providing a flexible interface with judiciously chosen CMS objects and appropriate NMT configuration data.

CAN-BIGBOX Communication Processor

A natural candidate for a module having the characteristics described here would be a communication processor attached to a S5 PLC via the S5 bus and connected to a CAN network with CAL. An implementation for a different setting is given with the CAN-BIGBOX.
This is an intelligent CAN field node which can communicate with up to 4 S5 PLCs using the RK512 protocol over a serial line (3964R). The PLCs may actually be remotely located and connected to the CAN-BIGBOX via modem and telephone lines. RK512 protocol FETCH requests are issued to the remote S5 for updating the process image in the CAN-BIGBOX cyclically. Clearly this is not an application with hard real time constraints. However the CMS modelling for this device is essentially as explained above.

References:

1) CAN Application Layer (CAL). CiA Draft Standards, 1994.
2) Berger, H.: Automatisieren mit SIMATIC S5-135U. Siemens, 1992.