DESIGN AND IMPLEMENTATION OF APPLICATION PROGRAM MODULE FOR A POWER STATION SCADA SYSTEM

Assmaa A. Fahad
Lecturer
Baghdad University / College of Science / Computer Science Department

ABSTRACT

Supervisory Control And Data Acquisition (SCADA) is a commonly used industry term for computer based system allowing system operators to obtain real-time data related to the status of an electric power system over wide geographic area.the designed SCADA system consists of Five modules and it is designed to work with two layers: Client layer and Data server layer each in a separated PC. The APplication Program (APP) module is one of the SCADA modules that works as an interface between the I/O cards and the other SCADA modules. It collects the data from the I/O cards; through an I/O interface card; and passes it to the Process Data Interchange (PDI) module. And because of the APP module works in a separated PC, it communicate with PDI module using the PC serial port and High-level Data Link Control (HDLC) communication protocol. The designed SCADA modules are programmed as a multithread programs written using Visual C++ programming language. The APP module is built to consist of three threads: one for collecting data, another to perform the communication operation, and the third thread is responsible of controlling all APP operations. The overall SCADA system including the APP module is implemented to supervise the operation of a power station since 1999 and it proves a very fast response time and provides a good real-time reports.

الخلاصة

يعتبر نظام السيطرة الإشرافية وجمع البيانات (SCADA) من الأنظمة الشائعة الاستخدام للسيطرة على المعامل التي تعتمد الحاسبات الشخصية لتجميع المعلومات عن محطات الكهرباء بالوقت الحقيقي وعبر مسافات بعيدة. يتكون نظام SCADA ألذي تم بناءه من خمس وحدات رئيسية تعمل بطبقتين برمجية، طبقة الموكل وطبقة محضر البيانات وعلى حاسبتين منفصلتين. وحدة البرامج التطبيقية هي إحدى وحدات النظام الذي يعمل كواجهة بين حقول وحدات الإدخال والاخراج وباقي وحدات النظام. فهذه الوحدة مسؤولة عن عملية جمع البيانات من حقول وحدات الإدخال والاخراج وتمريرها إلى باقي الوحدات عن طريق وحدة تبادل المعلومات. ونظرا لعمل هذه الوحدة على حاسبة منفصلة فانها تستخدم وحدات الإدخال المتسلسل المتوفرة في الحاسبة كوسيلة اتصال مع وحدة تبادل المعلومات وباستخدام نظام التخاطب HDLC. تم برمجة وحدات نظام SCADA لتتكون من عدة خيوط برمجية باستخدام لغة

البرمجة ++Visual c . وحدة البرامج التطبيقية تم بناء برامجها لتتكون من ثلاث خيوط برمجية خصص أحدها لقراءة البيانات من وحدات الإدخال والإخراج فيما خصص الثاني للأشراف على عملية التخاطب وتبادل المعلومات من باقي وحدات النظام . أما الثالث وهو الرئيسي فقد برمج للسيطرة على عمل وحدات النظام ككل.

تم فحص عمل النظام بجميع وحداته للإشراف على عمل إحدى محطات الكهرباء منذ العام ١٩٩٩ وقد اثبت كفاءة عالية في الأداء من حيث وقت الاستجابة للإشارات الداخلة ونوعية التقارير التي تم تحريرها.

KEY WORDS:

APP module: APplication Program module
FCS: Frame Check Sequence
SCADA: Supervisory Control And Data Acquisition
CRC: Cyclic Redundancy Check
HDLC: High-level Data Link Control

INTRODUCTION

SCADA systems have made substantial progress over the recent years in terms of functionality, scalability, performance and openness such that they are an alternative to in house development even for very demanding and complex control systems as those of physics experiments.

SCADA system is not a full control system, but rather focuses on the supervisory level. As such, it is a purely software package that is positioned on top of hardware to which it is interfaced, in general via Programmable Logic Controllers (PLCs), or other commercial hardware modules. SCADA systems are used not only in industrial processes: such as steel making, power generation (conventional and nuclear) and distribution, chemistry, but also in some experimental facilities such as nuclear fusion. The size of such plants range from a few 1000 to several 10 thousands I/O channels. However, SCADA systems evolve rapidly and are now penetrating the market of plants with a number of I/O channels of several 100 K [Dennise 1987].

SCADA SYSTEM ARCHITECTURE

There are two basic layers in a SCADA system: the "client layer", which caters for the man machine interaction, and the "data server layer" which handles most of the process data control activities. The data servers communicate with devices in the field through process controllers. Process controllers, e.g. PLCs, are connected to the data server either directly or via networks or field buses [Dennise 1987] .

The data server polls the controller in an operation called scanning operation. Scanning operation is performed at a user defined polling rate which is differing for different parameters. The controller pass the requested parameters to the data server, time stamping of the process parameters is typically performed in the controller and this time-stamp is taken over by the data server.

Server-client communication is in general based on a publish-subscribe and event-driven using one of common communication protocols [Ghanim 1999]. The client application subscribes to a parameter which is owned by a server application and only changes to that parameter are then communicated to the client application.

The SCADA system software is multi-tasking software based upon a real-time database located in the server. Normally the SCADA system is built to consist of a number of modules each responsible of a specific function. The number of these modules and the functions they performed are depending on the requirements of the system the SCADA is built to control.

SCADA SYSTEM FUNCTIONS

The SCADA systems are designed to perform the following functions [Ghanim 1999]:

- **Access Control**

  Users are allocated to groups, which have defined read/write access privileges to the process parameters in the system and often also to specific product functionality.

- **Man Machine Interface (MMI)**

  The products support multiple screens, which can contain combinations of synoptic diagrams and text. They also support the concept of a "generic" graphical object with links to process variables. These objects can be "dragged and dropped" from a library and included into a synoptic diagram.

  Most of the SCADA products that were evaluated decompose the process in "atomic" parameters (e.g. a power supply current, its maximum value, its on/off status, etc.) to which a Tag-name is associated. The Tag-names used to link graphical objects to devices can be edited as required. The products include a library of standard graphical symbols, many of which would however not be applicable to the type of applications encountered in the experimental physics community.

  Standard windows editing facilities are provided: zooming, re-sizing, scrolling, etc. On-line configuration and customization of the MMI is possible for users with the appropriate privileges. Links can be created between display pages to navigate from one view to another.

- **Trending**

  The products all provide trending facilities (the parameters to be trended in a specific chart, number of trended parameters in each chart, etc). The trending feature is either provided as a separate module or as a graphical object (ActiveX), which can then be embedded into a synoptic display. XY and other statistical analysis plots are generally not provided.

- **Alarm Handling**

  Alarm handling is based on limit and status checking and performed in the data servers. More complicated expressions (using arithmetic or logical expressions) can be developed by creating derived parameters on which status or limit checking is then performed.

- **Logging/Archiving**

  The terms logging and archiving are often used to describe the same facility. However, logging can be thought of as medium-term storage of data on disk, whereas archiving is long-term storage of data either on disk or on another permanent storage medium. Logging is typically performed on a cyclic basis, i.e., once a certain file size, time period or number of points is reached the data is overwritten. Logging of data can be performed at a set frequency, or only initiated if the value changes or when a specific predefined event occurs. Logged data can be transferred to an archive once the log is full. The logged data is time-stamped and can be filtered when viewed by a user.

- **Report Generation**

  One can produce reports using SQL type queries to the archive, Real Time Data Base (RTDB) or logs. Although it is sometimes possible to embed EXCEL charts in the report, a "cut and paste" capability is in general not provided. Facilities exist to be able to automatically generate, print and archive reports.

.

THE DESIGNED POWER STATION SCADA SYSTEM

The System Architecture

The system is constructed around two PCs, the central computer and the I/O processor. The central computer PC is the client of the SCADA system, while the I/O processor PC is the data server. The I/O processor is connected to the process equipment by means of an I/O interface which provides interfacing between I/O cards bus line of the existing equipments and the bus of the I/O processor PC.

The designed system major parts are hardware I/O interface card and software modules that are designed to fulfill the system requirements. The software modules are:

- APplication Program module (APP).
- Process Data Interchange module (PDI).
- Data Base module (DB).
- Event Processing module (EP).
- Man-Machine Interface module (MMI).

Digital and analog data are acquired through the system hardware via the I/O interface driven by the I/O drivers in the APP module. The APP module residing in the I/O processor PC transmits the data to the PDI module in central PC via serial link using HDLC communication protocol.

PDI module manages the data exchange and transmission . After receiving the information, the PDI updates the data base files and transmits a transaction to the EP module. This occurs upon status change of digital data or if analog values exceed predefined limit, or in case of error occurrence in the system.

DB module manages the data base files and the logging and archiving process. The EP module is responsible of alarm handling and records the events to the operator.

MMI module display the information stored in the data base files either as diagrams or event lists or curves. MMI also provides initiation of user inquiries and commands, such as system diagnostic and test requests. Event lists, reports and other messages can be directed to a printer if required.

The scope of this paper is to design APP module in SCADA system.

The Designed APP Module

As mentioned before the APP module is responsible of providing the plant's status to another modules. Therefore it is structured to perform the following functions:

- Provide I/O driver routines for I/O interface card.
- Perform scanning operation .
- Interrupts handling.
- Transmit field status to central PC.
- Perform Diagnostic operation.

I/O Driver Routines

Driver routines for I/O interface card are required as a major part of any APP module. To achieve highest accuracy the structure of these drivers should be such fast response time and minimum software overhead. Therefore these programs should be simple, flexible and with minimum code.

To write I/O driver routines the structure of the I/O field and the data types this I/O field provides must be specified.

The designed SCADA system is implemented in a hardware with six thermal units connected to the computers. Inputs from the plant are interfaced directly to seven separated subsystems for analog inputs and digital inputs collection. One subsystem is foreseen for each of the six power units, and one dedicated to common services data collection[Baiji Manual 1989].

Several I/O driver routines are written to read different types of data. The system accepts plant data in form of analog inputs, digital inputs or pulse inputs and generates output digital signals to perform loop-back test to the I/O cards.

For analog inputs the plant includes 14 boards of 16 channels each. These inputs are applied to an Analog to Digital Converter (ADC) via multiplexers. At the end of conversion the

ADC generates an interrupt signal after which APP module can read the analog data from ADC buffer.

The digital inputs arranged in 32 boards of 16 channels each. These channels are known as Digital-scan channels. Two driver routines are written to read data from these channels, one to read all 32 boards at one time and another to read single specified board.

Pulse input channels are digital input with interrupt, changing the status of any channel will cause an interrupt signal; these channels are known as Digital-interrupt channels. The signals connected to these channels are with the highest priority in the system. In this type the inputs arranged in 16 boards of 12 channels each. The data from these channels are reads either when an interrupt occurs from these channels or by using an I/O driver routine written to allow selective reading of these channels at normal time (no interrupt).

Scanning Operation

In this operation the APP module reads the status of the plant from the I/O field. At the initialization of SCADA system all channels are scanned in order to record the initial state of the plant. After that the scanning operation is performed periodically depending on the rate of change of data on these channels. For the designed system and depending on this rate of change the analog channels are divided into seven types: one second, two seconds, five seconds, ten seconds, twenty seconds, thirty seconds, and sixty seconds data change rate channels. In accordance with this classification the APP module perform the scanning operation, some channels scanned every one second another scanned every two seconds and so on. Digital–scan input channels are all with the same data rate change, every one second, so they are all scanned every one second. Digital-interrupt channels are not scanned for there nature of operation. All channels are scanned, but only the changed one will transmitted to central PC.

During the scanning operation, the APP module reads the status of the channel and compare it with the old one stored in the DB, if there is a change the channel address, the new status of the channel, and the time stamp (if need) are stored in a circular queue in order to be arranged in a frame and send to central PC.

Three circular queues are defined to store the status of the channels one for each type (Digital-scan, Digital-interrupt, and Analog). Because of APP module is built as a multithread program these queues may accessed from more than one thread at the same time. To synchronize the accessing of these queues without any error, these queues are defined as a critical section area [Silberschatz 1998].

With Digital-scan channels and analog channels there is no needs for a time stamp because they scanned periodically, therefore the status of each channel is represented in only three bytes (one for channel address and two bytes for channel status in its digital form). With digital-interrupt channels it is very important to fix the time of the event, therefore the status of each channel will represented in five bytes (6-bit channel address, 4-bit modified point in this channel, 1-bit for point status, 1-bit not used, and 28-bit for time stamp which is fixed in millisecond), Fig.(1).

<table>
<tr><td colspan="2">0</td><td></td><td>7</td></tr>
<tr><td colspan="2">Channel Address (6-bit)</td><td>Point status (1-bit)</td><td>not used (1-bit)</td></tr>
<tr><td>Modified Digital Point No. (4-bit)</td><td colspan="3">Millisecond (4-bit)</td></tr>
<tr><td colspan="4">Millisecond (8-bit)</td></tr>
<tr><td colspan="4">Millisecond (8-bit)</td></tr>
<tr><td colspan="4">Millisecond (8-bit)</td></tr>
</table>

Fig (1) Digital-interrupt information record

Interrupt Handling

One of the most important and critical operation needs to be handled by APP module is handling the interrupt signals generated by the hardware components. The designed APP module handles four interrupt signals:

- Interrupt signal from A/D converter: When A/D completes his operation it sends an interrupt signal, the Interrupt Service Routine (ISR) handles this interrupt signal will read the data from A/D buffer and initiate, if any, next analog scanning.
- Interrupt signal from system timer: The APP module programs Intel 8254 system timer to generate an interrupt signal every 150ms. Accordingly the APP will update the time and the date of the system. The APP module use this time to synchronize the communication operation with central PC, fix the time of scanning operation, record the time of the events, and reset the watch dog hardware timer. The watch dog timer is a part of the I/O interface card that used to insure continuous operation of the system. This timer must resets every 200ms, if for some reason the system is stopped and the watch dog timer not resets, this timer will cutoff the power supply of the I/O boards.
- Interrupt signal from any changed Digital-interrupt channel: The ISR handles this interrupt signal will read the new status of this channel and records it, with all other information, in a digital-interrupt circular queue.
- Interrupt signal from serial port: PC serial port is used to communicate the data with central PC. It is programmed to send an interrupt signal when the receiver buffer is full. The ISR reads the characters in the buffer and return the control to APP main routine.

Transmit Field Status to Central PC

This part of APP module is designed to transmit the status of the channels from APP module to central PC through PDI module. The two PCs, I/O processor PC and central PC are placed in two different places and they are connected through RS232 serial port. HDLC communication protocol is used to control the communication operation between these PCs [Halsall 1996].

Different types of messages are defined to handle the communication operation some of these messages are used to control the communication operation and another are used to hold data. In HDLC protocol both data and control messages are carried in a standard frame format, Fig (2) [Halsall 1996].

Start of frame	Frame Header (16 bit)		Information Field	Frame Check	End of frame
delimiter (8-bit)	8-bit	8-bit	0.. N	Seq. (16-bit)	delimiter (8-bit)
Flag	Address	Control	Information	FCS	Flag

Fig (2) Standard HDLC frame format

The designed SCADA system uses the Flag byte to define the start and the end of the message. The address field is used to check for data transparency (the appearance of flag sequence in frame contents) because the connection between the two PCs is point-to-point [Halsall 1996].

The Control field is used to define the frame type: Unnumbered frame, Information frame, and Supervisory frame. For each type the designed SCADA system use the control field as in Fig (3).

	1	2	3	4	5	6	7	8
Information frame	0	N(S)			P/F	N(R)		

Where N(S) : send sequence number
N(R) : receive sequence number
P/F : Poll / Final bit

	1	2	3	4	5	6	7	8
Supervisory frame	1	0	S		P/F	N(R)		

Where S : Receiver Ready – RR
Receiver Not Ready - RNR
Reject – REJ
Selective Reject – SREJ

	1	2	3	4	5	6	7	8
Unnumbered frame	1	1	M		P/F	M		

Where M : Unnumbered commands
Set Normal Response Mode (SNRM)
Frame Reject (FRMR)
Disconnect (DISC)
Unnumbered responses
Unnumbered Acknowledge (UA)
Frame Reject (FRMR)
Disconnected Mode (DM)

Fig (3) SCADA system HDLC Control Field formats

Frame Check Sequence (FCS) field is a 16-bit Cyclic Redundancy Check (CRC) computed for the complete frame contents enclosed between the two flag delimiters. This field is computed and transmitted with each message in order to detect any error may occur during the communication operation. The CRC error detection method is used to increase the reliability of the system [Halsall 1996].

These information messages and control messages are implemented in both APP and PDI modules. The APP module receives a number of control messages from PDI module that control the communication operation between the two PCs. Five link modes are defined in communication operation: Link-establishment mode, Time-sync mode, Data-request mode, Diagnostic mode, and Link-disconnected mode.

The data communication operation is activated when the link is established upon request from central PC. When the link is established between the two PCs a time-sync mode is activated in which central PC sends a time-sync message to synchronize the time in two PCs. When APP receives a time-sync message it updates the timer counters in I/O processor PC accordingly. Time-sync mode is activated from time to time to always insure the synchronization of the time between the two PCs.

Data-request mode is activated after time-sync mode, in this mode APP module begins a transmission of data messages. One information message can hold the status of more than one channel but with the same type, for example one message can hold the status of all changed Digital-scan input channels that scanned at specific time. With any transmission error, APP module retransmits the last message for three times and if there is still an error a Link-establishment mode is reactivated to check the status of the connection between the two PCs. With any error occurs during the communication or scanning operation the APP module also sends an error message such as: Dig-scan-buffer-full, Address-error-scan-seq., Analog-buffer-full, etc to central PC in order to recover it, Fig (4).

8-bit	8-bit	Control				1-bit	10-bit	5-bit	16-bit	8-bit
Flag	Address	0	N(S)	P/F	N(R)	Error	Channel Address	Error code	CRC	Flag

Fig (4) SCADA system error message format

After sending any error message the APP module continues with his normal operation. The PDI module passes this error message to EP module in order to display a suitable report to the operator describing the problem in the system.

Diagnostic Operation

Checking the performance of the I/O boards in the I/O field is called diagnostic operation. Diagnostic operation is performed by using special diagnostic cards. These cards are used to route a defined pattern of zeros and ones to the I/O boards. The APP then reads the data from the I/O boards and compares it with the written one. The diagnostic operation is activated by central PC when there are a number of error messages (such as Address-error-scan-seq.) are received from APP module.

When APP module receives a diagnostic request message from central PC it suspends all the operations and changes the mode of operation to diagnostic mode and begins the diagnostic operation. After completing the diagnostic operation the APP module sends the diagnostic results in messages to central PC and resumes his operations. The EP module in central PC will display a report about the diagnostic result so the operator can make a decision about the system accordingly.

DISCUSSION AND CONCLUSION

In spite of the complexity of the SCADA system and very large and different types of signals received from the I/O fields, the designed SCADA system improves a good performance when it is implemented to supervise the operation of a power station since 1999. The system performance measure is by examining the CPU usage and the memory usage. During the operation of the designed SCADA system the PCs are work with minimum CPU and memory usage.

Programming SCADA modules as a multithread system simplify running these modules concurrently and simplify the operation of writing very efficient programs with maximum use of the CPU.

HDLC communication protocol is very suitable communication protocol for such SCADA system for his flexible frame format that makes the system provides different types of control and information messages and a different number of operation modes that help the programmer to cover all the requirements of the system.

REFERENCES

"Data logger and alarm system for Baiji power station", Technical Manual, 1989.

Dennise J. G. and Henry T. D. " Supervisory Control and Data Acquisition", proceeding of the IEEE, Vol.7, No. 12, December 1987.

Ghanim, Z. N. " Design and Implementation of Small SCADA System", a master theses, Baghdad University, College of Engineering, 1999.

Halsall, F. " Data Communications, Computer Networks and Open System", fourth edition, Addison Wesley, 1996.

Silberschatz A. and Galvin P. B. "Operating system concepts", Fifth edition, Addison-Wesley, 1998.