A Sierra Monitor Company

Driver Manual
(Supplement to the FieldServer Instruction Manual)

FS-8700-79 GCM_ASCII Driver
Suitable for connection to a Barber Coleman GCM unit.

APPLICABILITY & EFFECTIVITY

Effective for all systems manufactured after May 1, 2001

TABLE OF CONTENTS

1 GCM ASCII Description

The GCM_ASCII driver allows the FieldServer to transfer data to and from devices over either RS232 or RS485 using ASCII.

GCM is an acronym for Global Control Module.
LCM is an acronym for Local Control Module.

The driver may only be configured as a client. Limited server side functionality has been implemented for QA test procedures but it is not supported or documented. If you require server side functionality, please contact Field Server's marketing group.

The GCM ASCII Serial Driver allows variables to be read and written in devices connected as Barber Coleman Network 8000. GCM reports for ZONE2 block types and Current status for LCMs can be read and blocks parameters for GCM itself and LCM can be overriden

The GCM devices provides a gateway to read/write data to its own blocks and blocks owned by other devices(LCMs,Microzones etc,) connected to Global Control Module. This driver polls the Global Control Module in network 8000 which in turn reads / writes data to itself or other connected devices.

The driver is an active client driver. This means that it initiates read / write polls with the GCM (Barber-Coleman Network 8000) GCM device, which is expected to provide responses. Server functionality is not provided by in this driver. but it can be extended on request and typically at additional cost.

The driver can be configured to allow data blocks to be read from GCM and/or other devices connected to GCM. As the blocks typically contain more than one data element, the retrieved data is stored in a number of consecutive FieldServer data arrays and locations in the Field Server.

The driver can be configured to read a specific variable and store its value in a configurable location in a FieldServer data array.

The driver has no advanced knowledge of the GCM or other Network devices and their data blocks. Therefore it cannot validate block or variables names specified in the configuration file. In addition, this means that the driver handles each variable/block in a generic way and without regard for the particular variables that constitute the tables.

It is important that you understand the limitations and exclusions of this driver. Read section 7.1 for more information.

2 Driver Scope of Supply

2.1 Supplied by FieldServer Technologies for this driver

FieldServer Technologies PART #	DESCRIPTION
FS-8915-10	UTP cable (7 foot) for RS232 use
FS-8917-01	RJ45 to DB25M connection adapter
	Driver Manual.

2.2 Provided by user

PART #	DESCRIPTION
	GCM from Barber-Coleman network 8000

3 Hardware Connections

The bridge is connected to the Global Control module as shown below.

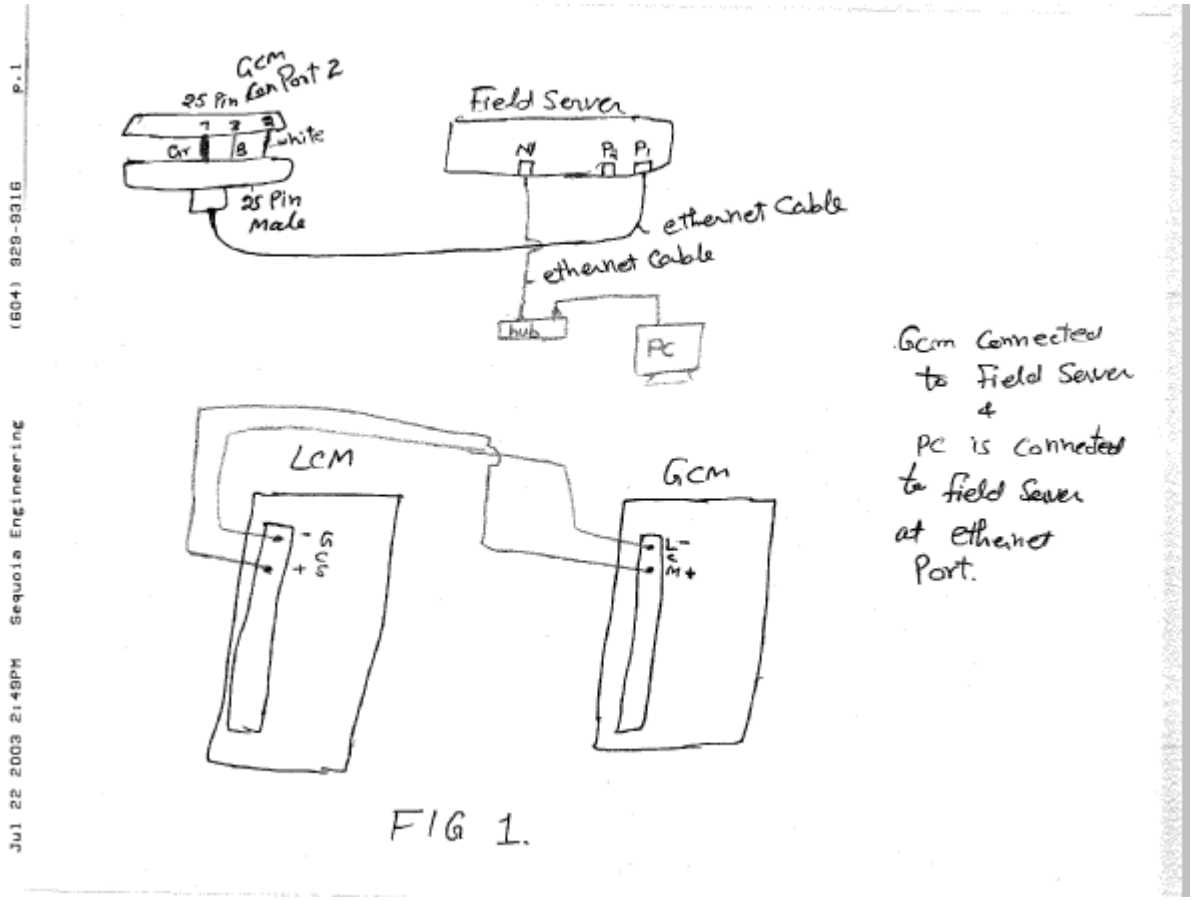

FIG 1.

4 Configuring the FieldServer as a GCM_ASC_Client

It is not possible to complete a configuration for communication with a GCM device until you are familiar with the data available from the devices connected to N8000 network. The GCM device does provide a method for discovering the data blocks and variables that are available in all the connected devices.

Configuring the client require minimum knowledge of the names of the data blocks available in the devices connected via the N8000 network to GCM. If you know the block names and the variable names and all the menu options to get to the required block or variable that you wish to poll then you have enough information to complete the configuration. If you have do not have block, variable and other menu options we provide a method of discovering this information. This method is discussed in chapter 7.

For a detailed discussion on FieldServer configuration, please refer to the instruction manual for the FieldServer. The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer (See ".csv" files on the driver diskette).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with a GCM device.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for GCM communications, the driver independent FieldServer buffers need to be declared in the "Data Arrays" section, the destination device addresses need to be declared in the "Client Side Nodes" section, and the data required from the servers needs to be mapped in the "Client Side Map Descriptors" section. Details on how to do this can be found below.

Note that in the tables, * indicates an optional parameter, with the bold legal value being the default.

4.1 Data Arrays

Section Title
Data_Arrays

Column Title	Function	Legal Values
Data_Array_Name	Provide name for Data Array	Up to 15 alphanumeric characters
Data_Format	Provide data format. Each data array can only take on one format.	FLOAT, BIT, UInt16, SInt16, Packed_Bit, Byte, Packed_Byte, Swapped_Byte
Data_Array_Length	Number of Data Objects. Must be larger than the data storage area required for the data being placed in this array.	1-10,000

Example

```
  //    Data Arrays
  //
Data_Arrays
Data_Array_Name  ,Data_Format  ,Data_Array_Length
DA_Val           ,float        ,2
DA_Name          ,Byte         ,44
DA_Typ           ,Byte         ,2
DA_Units         ,Byte         ,32
```

4.2

Here DA_Val data array is declared to store float values for variables and length (2) indicates that this data array can store at most values for two variables.

DA_Name data array is to store Name of the variables. Twenty two bytes are reserved for each variable name so length 44 indicates that this data array can contain at most two variable names.

DA_Typ data array is declared to store numeric value for variable types. Single byte can represent the variable type ,so length 2 indicate that this data can hold variable types for at most two variables. See the table for corresponding numeric values for types in chapter 5.

DA_Units data array is declared to store units for variables. Again 16 bytes are reserved to store units for each variables and length (2) indicates that it can store units for at most two variables.

4.3 Client Side Connections

Section Title		
Connections		
Column Title	**Function**	**Legal Values**
Port	Specify which port the device is connected to the Bridge	P1-P8, (R1-R2 with converter)
Baud*	Specify baud rate Driver supports all standard baud rates between 110 – 115200. Vendor equipment is limited.	9600
Parity*	Specify parity Driver supports: Even, Odd, **None,** Mark, Space. Vendor equipment is limited.	**None**
Data_Bits*	Specify data bits Driver supports 7,8. Vendor equipment is limited.	**8**
Stop_Bits*	Specify stop bits Driver supports 1,2. Vendor equipment is limited.	**1**
Protocol	Specify protocol used	GCM_asc
Handshaking*	Specify hardware handshaking	**None**
Poll _Delay*	Time between internal polls This driver does not require a poll delay.	0-32000 seconds **default 1 second**
Timeout*	Please adjust this parameter to 5.0s for proper functioning.	5.0s

Example

```
//    Client Side Connections

Connections
Port, Baud, Parity, Data_bits, Stop_Bits, Protocol , Handshaking,
P8,   9600, None  , 8        , 1        , GCM_asc  , None        ,
```

4.4 Client Side Nodes

Section Title		
Nodes		
Column Title	**Function**	**Legal Values**
Node_Name	Provide unique name for node	Up to 32 alphanumeric characters
Node_ID	This commonly used field has no meaning for this driver.	
Protocol	Specify protocol used	GCM_Asc
Port	Specify which port the device is connected to the Field Server.	P1-P8, R1-R2
User_name	User id to enter in GCM	Up to 16 character or numeric
Password1	Password to enter in GCM	Up to 16 character or numeric
Dev_type	Dev type information from GCM's Version info	GCM or NSIM More info in section "7 - Connecting through Hyper Terminal"

Example

```
//    Client Side Nodes

Nodes
Node_Name, Node_ID, Protocol , Port, User_name ,Password1,dev_type
GCM_01    , 1      , GCM_Asc , P1  , user      ,pass     ,NSIM
```

4.5 Client Side Map Descriptors

4.5.1 FieldServer Related Map Descriptor Parameters

Column Title	Function	Legal Values
Map_Descriptor_Name	Name of this Map Descriptor	Up to 32 alphanumeric characters
Data_Array_Name	Name of Data Array where data is to be stored in the Bridge	One of the Data Array names from "Data Array" section above
Data_Array_Offset	Starting location in Data Array	0 to maximum specified in "Data Array" section above
Da_Byte_Name*	Name of Data Array where Name of the variable is to be stored in the Bridge	One of the Data Array names from "Data Array" section above
Da_Float_Name*	Name of Data Array where Type of variable is to be stored in the Bridge	One of the Data Array names from "Data Array" section above
Da_Bit_Name*	Name of Data Array where units for variable is to be stored in the Bridge	One of the Data Array names from "Data Array" section above
Function	Function of Client Map Descriptor	RDBC, WRBC, WRBX

4.5.2 Driver Related Map Descriptor Parameters

Column Title	Function	Legal Values
Node_Name	Name of Node to fetch data from	One of the node names specified in "Client Node Descriptor" above
Data_Type	Data type This commonly used parameter is not required for this driver	-
Length	Length of Map Descriptor. Or number of variables to read Keep this value always 1 then driver can handle itself while reading one parameter or all parameters of Block Data report or Current Status.	1
Address	This commonly used parameter is not required for this driver.	-
GCM_Main_Menu	The GCM's Main Menu Option to be browsed. Eg. REPORTS	You will discover the value for this by using the method provided in section "7 - Connecting through Hyper Terminal"
GCM_Sub_Menu	The GCM's Sub Menu Option	You will discover value for this

	to browse under Main Options. Eg. BLOCK DATA Is a sub option under REPORTS menu.	by using the method provided in section “7 - Connecting through Hyper Terminal”
LCM_Name*	This is name of the LCM connected to GCM. Skip this parameter if you do not wish to communicate with LCM . Read chapter 7 for more information.	You will discover value for this by using the method provided in section “7 - Connecting through Hyper Terminal”
User_Name	This is a userid to access the LCM. Skip this parameter if you do not wish to communicate with LCM.	You will discover value for this by using the method provided in section “7 - Connecting through Hyper Terminal”
Password1	This is a password to access the LCM. Skip this parameter if you do not wish to communicate with LCM.	You will discover value for this by using the method provided in section “7 - Connecting through Hyper Terminal”
Lcm_Main_Menu*	Similar to GCM_Main_Menu But can skipped if LCM communication is not required	You will discover value for this by using the method provided in section “7 - Connecting through Hyper Terminal”.
LCM_Sub_Menu*	GCM_Sub_Menu But can skipped if LCM communication is not required or if this sub menu option is not relevant (eg to read LCM Current Status, this option is irrelevant.)	You will discover value for this by using the method provided in section “7 - Connecting through Hyper Terminal”.
Block_Name*	This is a name of the block that may contain one or more variables. Eg. ZONE2:TOO is full block name. Here ZONE2 again is the *type* and TOO is name of the *block* . But provide full name to read a report for data block for microzone II.	You will discover value for this by using the method provided in section “7 - Connecting through Hyper Terminal”

Variable_Name*	This is variable name. This parameter can exist without it Block_Name parameter. Eg. AO:SSS2 the is considered a variable name. Actually AO is the *type* and SSS2 is a *name* of the block. Mention full name like this to override in LCM. And AO:CWVLV:AV is a complete variable name to read from Current status of LCM. Here AO is type CWVLV is name and AV is attribute for variable. UIAV1 forms a variable complete name for MicrozoneII	This parameter should be skipped all parameter should be read from LCM Current Status or from Block Data Report for MicrozoneII. You will discover value for this by using the method provided in section "7 - Connecting through Hyper Terminal"

4.5.3 Timing Parameters

Column Title	Function	Legal Values
Scan_Interval	Rate at which data is polled	>0.1s
Timeout	This is important parameter but must be kept constant to 5s	5s

4.5.4 Map Descriptor Example 1 – Read All Current Status Variables for LCM

This example illustrates how to read the Current Status for all variables in LCM. Some basics first: The map descriptor is connected to a node by means of the node name and a node definition provides a connection to a port. Thus this map descriptor is connected to a port via its node. The FieldServer will use that port to send this poll. The poll will be generated every 0.5 seconds after finishing with previous poll.

The value extracted from the response will be stored in the array called DA_Val.
Variable name will be stored in data array DA_Name.
Variable type will be stored in data array DA_Typ.
Variable units will be stored in data array DA_Units.
Ensure that it is suitable format for storing value – FLOAT is suggested.
Driver will store information in the following manar.
For first variable
Variable value will be stored at 0th offset at Da_Val data array.
Variable name will be stored at 0th offset through 21st offset at Da_Name data array.
Variable type will be stored at 0th offset at Da_typ data array
Variable units will be stored at 0th offset through 15th offset at Da_Units data array.

For Second variable
Variable value will be stored at 1th offset at Da_Val data array.
Variable name will be stored at 22nd offset through 43 rd offset at Da_Name data array.
Variable type will be stored at 1th offset at Da_typ data array
Variable units will be stored at 16th offset through 31th offset at Da_Units data array.
And so on…..

Map Descriptors

```
Map_Descriptor_Name ,scan_interval ,Data_Array_Name ,Data_Array_Offset ,Da_Byte_Name ,Da_Float_Name ,Da_Bit_Name  ,Function  ,Node_Name  ,GCM_Main_Menu
CLCM_1              ,0.5s          ,DA_Val          ,0                 ,DA_Name       ,DA_Typ        ,DA_Units      ,Rdbc      ,GCM_01     ,ACCESS
```

Unique name Give any name like mydesc_1, Mysesc_2 etc

Indicates the data array name that will store the values for variable s for entire report

Starting position within data array to store values for this map desc.

These three are data array names declared in data array section to store Name, type and units for variables.

The driver generates a read poll continuously.

The node name connects the map descriptor to a node definition.

One of the options available at Main Menu of GCM.

Mapdesc explanation continue……..

```
,GCM_Sub_Menu  ,LCM_Name          ,User_name  ,Password1  ,Lcm_Main_Menu  ,Length  ,Timeout
,LOCAL_LCM     ,LCM_2:OCC_AH20    ,user       ,pass       ,Current Status  ,1       ,5s
```

One of options available after selecting main option

Name of LCM from which Current Status report should be read.

Usre name and password to access LCM.

Again this is the one of the options available at Main menu of Lcm.

Keep Length always 1, driver can handle itself to number of successfully read variables.

Keep Timout to 5s to avoid malfunctioning .

4.5.5 Map Descriptor Example 2 – Read a Single Current Status Variable for LCM

This example illustrates how to read a single variable from Current Status of LCM. Make one similar map descriptor for each variable you wish to read. Reading variables is a more efficient you want to read a limited set of variables as per as data array memory is concerned.

Map Descriptors

```
Map_Descriptor_Name ,scan_interval ,Data_Array_Name ,Data_Array_Offset ,Da_Byte_Name ,Da_Float_Name ,Da_Bit_Name  ,Function  ,Node_Name  ,GCM_Main_Menu
CLCM_1              ,0.5s          ,DA_Val         ,0                  ,DA_Name      ,DA_Typ        ,DA_Units      ,Rdbc      ,GCM_01      ,ACCESS
CLCM_2              ,0.5s          ,DA_Val         ,1                  ,DA_Name      ,DA_Typ        ,DA_Units      ,Rdbc      ,GCM_01      ,ACCESS
```

```
,GCM_Sub_Menu  ,LCM_Name           ,User_name  ,Password1  ,Lcm_Main_Menu  ,Variable_Name          ,Length  ,Timeout
,LOCAL LCM     ,LCM  2:OCC AH20    ,user       ,pass       ,Current Status  ,AO:CWVLV:AV           ,1       ,5s
,LOCAL LCM     ,LCM  2:OCC AH20    ,user       ,pass       ,Current Status  ,LOOP:HTGCLG:CALSP     ,1       ,5s
```

As we are using the same data array for both variables, we change the offset.

The value for 'AO:CWVLV:AV ' will be stored at index 0 (first element), the value for LOOP:HTGCLG:CALSP will be stored at index 1 (2nd element of array).

AO:CWVLV:AV and ,LOOP:HTGCLG:CALSP are the Variables name for which driver will read information.

4.5.6 Map Descriptor Example 3 – Override (Write) a Variable at LCM

This example illustrates how to Override a variable. Always set the length to 1 for a write.
In this example the variable being overridden is called 'AO:SSS2'. It can be set to any analog value but Lcm may have range for physical points (eg 0.0- 12.0)..

```
Map_Descriptors
Map_Descriptor_Name ,scan_interval ,Data_Array_Name ,Data_Array_Offset ,Function  ,Node_Name  ,GCM_Main_Menu  ,GCM_Sub_Menu ,LCM_Name
CLCM_3              ,0.5s          ,DA_Val3         ,0                 ,wrbc      ,GCM_01     ,ACCESS          ,LOCAL LCM    ,LCM  2:OCC AH20

 ,User_name ,Password1  ,Lcm_Main_Menu  ,LCM_Sub_Menu  ,Variable_Name  ,Length  ,Timeout
 ,user      ,pass       ,BLOCKS         ,OVERRIDE      ,AO:SSS2        ,1       ,5s
```

This is the data array that should be declared in data array section. New value for override will be picked from this data array.

Again this is one of the options available under Lcm main menu.This function ensures the driver write the Setpoint to the device continuously (every 0.5 seconds in this example),
Only If this is only the map descriptor in configuration file.
Use WRBX to generate a write message each time the

Again this is one of the options available under Lcm Blocks menu. This is the operation that will be performed upon variable.

This function ensures the driver write the Setpoint to the device continuously (every 0.5 seconds in this example),
Only If this is only the map descriptor in configuration file.
Use WRBX to generate a write message each time the value in the array is updated.

This is the exact name of the variable going to be override.

4.5.7 Map Descriptor Example 4 – Read all Variables for Block representing Microzone II at GCM

This example illustrates how to read all variables from the block that describe the MicroZone II device at GCM.

```
Map_Descriptors
Map_Descriptor_Name ,scan_interval  ,Data_Array_Name ,Data_Array_Offset ,Da_Byte_Name ,Da_Float_Name ,Da_Bit_Name  ,Function  ,Node_Name  ,GCM_Main_Menu
CMD_3               ,0.5s           ,DA_Val          ,0                 ,DA_Name      ,DA_Typ        ,DA_Units     ,Rdbc      ,GCM_01     ,REPORTS

,GCM_Sub_Menu                ,Block_Name    ,Timeout
,BLOCK DATA                  ,ZONE2:TOO     ,5s
```

One of options available after selecting main option

This is the name of the block that represents the MciroZone II device at GCM. Report for this block will read.

One of the options available at Main Menu of GCM.

4.5.8 Map Descriptor Example 5 – Read Single Variable for Block type ZONE2 representing Microzone II at GCM.

This example illustrates how to read a single variable from ZONE:TOO block that contains data for Micro Zone II device. Make one similar map descriptor for each variable you wish to read. Reading variables is a more efficient if you want to read a limited set of variables as per as data array memory is concerned.

```
Map_Descriptors
Map_Descriptor_Name ,scan_interval ,Data_Array_Name ,Data_Array_Offset ,Da_Byte_Name ,Da_Float_Name ,Da_Bit_Name  ,Function  ,Node_Name  ,GCM_Main_Menu
CMD_1               ,0.5s          ,DA_Val         ,0                 ,DA_Name       ,DA_Typ        ,DA_Units     ,Rdbc      ,GCM_01     ,REPORTS
CMD_2               ,0.5s          ,DA_Val         ,1                 ,DA_Name       ,DA_Typ        ,DA_Units     ,Rdbc      ,GCM_01     ,REPORTS

,GCM_Sub_Menu   ,Block_Name      ,Variable_Name                ,Length ,Timeout
,BLOCK DATA     ,ZONE2:TOO       ,AO2                          ,1      ,5s
,BLOCK DATA     ,ZONE2:TOO       ,DV3                          1       ,5s
```

AO2and ,DV3 are the Variables name for which driver will read information from ZONE2:TOO block.

As we are using the same data array for both variables, we change the offset.

The value for 'AO2' will be stored at index 0 (first element), the value for DV3 will be stored at index 1 (2nd element of array).

4.6 Client Side Configuration File Examples with Notes

Black Italics – CSV file Data
Blue Italics - CSV file comments
Blue regular – Additional comments for this manual.

4.6.1 CSV Example 1 – Read Block Data from a GCM

The csv file provided below has been configured to read Micro Zone II type devices connected to GCM

```
//=============================================================================
//
//  1.00aA 30 May 03 SSS  Created
//
//=============================================================================*/


//=============================================================================
//
// Notes :  None.
//
//
//=============================================================================


//=============================================================================
//
//    Common Information
//
Bridge
Title
Gcm_Asc Testing Client


//=============================================================================
//
//    Data Arrays
//

// to read data for ZONE2:TOO from block data under report menu
```

```
// DA_Val---------> will contain the value of the parameter
// DA_Name--------> will contain the name of the parameter (each will occupy at least 22 bytes )
// DA_TYP---------> will contain the type parameter
// DA_Units-------> will contain the units for  value (each will occupy at least 16 bytes )

// NOTE: if "value" is not available 99 is considered as default value to show some progress during test
//       if "typ"   is not available 99 is considered as default typ to show some progress   during test
//      if "units" are not available "N/A" will stored, same will be with Digital values

// DA_TYP may have value one of the following if we found successfully
// Check with manual for most updated values
// AI                  1   // simple analog inputs
// AO                  2   // simple analog outputs
// DI                  3   //Digital Inputs
// DO                  4   // Digital outputs
// UI_A                5   // universal Input Analog
// UI_D                6   // universal Input Digital
// WINDOW_AV           7   // Window Analog  value(for micro zone II) READ ONLY
// WINDOW_DV           8   // Window Digital value(for micro zone II) READ ONLY
// EMS_AI              9   // Ems Analog  Value (for microzone II) Writable
// EMS_DI              10  // Ems Digital Value (for microzone II) Writable
// LOOP                11  // for LCM Current status

// These data arrays are used to read only two parameters under ZONE:TOO block
// for CMD_1 and CMD_2 mapdesc
Data_Arrays
Data_Array_Name  ,Data_Format  ,Data_Array_Length
DA_Val           ,float        ,2
DA_Name          ,Byte         ,44
DA_Typ           ,Byte         ,2
DA_Units         ,Byte         ,32



// these data arrays are used to read all parameters (36) under ZONE:TOO block
// for CMD_3 mapdesc
Data_Arrays
Data_Array_Name ,Data_Format  ,Data_Array_Length
DA_Val2         ,float        ,36
DA_Name2        ,Byte         ,792
DA_Typ2         ,Byte         ,36
DA_Units2       ,Byte         ,576




// these data arrays are used to read only two parameters under ZONE:TOO, but will contain only NAME and VALUE for parameters
// for CMD_4 , CMD5 mapdesc
Data_Arrays
Data_Array_Name ,Data_Format  ,Data_Array_Length
```

```
DA_Val3          ,float         ,2
DA_Name3         ,Byte          ,44



//=====================================================
//
//    client Side Connections
//

Connections
Port ,Baud ,Protocol ,Timeout
P1   ,9600 ,Gcm_Asc  ,5s


//=====================================================
//
//    client Side Nodes
//

Nodes
Node_Name ,Node_ID ,Protocol ,Port ,User_name ,Password1 ,dev_type
GCM_01    ,1       ,Gcm_Asc  ,P1   ,user      ,pass      ,NSIM


//
//    Client Side Map Descriptors
//
//reading maps
// MD CMD_1 & CMD_2 will read a parameter A02 and DV3 respectively from zone II type block, under block data option of REPORTS mneu
// and will store values in DA_Val array at offsets 0 and 1 respectively
//                types     DA_TYP                 0     1
//                Names     DA_Name                0     22  --------------(22 characters each)
//                units     DA_units               0     16  --------------(16 characters each)
Map_Descriptors
Map_Descriptor_Name ,scan_interval ,Data_Array_Name ,Data_Array_Offset ,Da_Byte_Name ,Da_Float_Name ,Da_Bit_Name  ,Function
,Node_Name  ,Gcm_Main_Menu  ,Gcm_Sub_Menu ,Block_Name   ,Variable_Name ,Length
CMD_1              ,0.5s          ,DA_Val          ,0                  ,DA_Name      ,DA_Typ        ,DA_Units     ,Rdbc      ,GCM_01
,REPORTS        ,BLOCK DATA   ,ZONE2:TOO    ,AO2           ,1      ,5s
CMD_2              ,0.5s          ,DA_Val          ,1                  ,DA_Name      ,DA_Typ        ,DA_Units     ,Rdbc      ,GCM_01
,REPORTS        ,BLOCK DATA   ,ZONE2:TOO    ,DV3           ,1

// MD CMD_3 will read all parameters from zone II type block, under block data option of REPORTS mneu
// and will store values in DA_Val array at offsets 0,1, 2 , ---onward
//                types     DA_TYP                 0,1, 2, ---onward
//                Names     DA_Name                0,22,44,66 --onward ----------(22 characters each)
//                units     DA_units               0,16,32,48---onward ----------(16 characters each)

Map_Descriptors
```

Always set the timeout to at least 5 seconds.

These are the password and id used to log into the GCM.

```
Map_Descriptor_Name ,scan_interval ,Data_Array_Name ,Data_Array_Offset ,Da_Byte_Name ,Da_Float_Name ,Da_Bit_Name  ,Function
,Node_Name  ,Gcm_Main_Menu  ,Gcm_Sub_Menu ,Block_Name
CMD_3               ,0.5s          ,DA_Val2         ,0                 ,DA_Name2     ,DA_Typ2       ,DA_Units2    ,Rdbc      ,GCM_01
,REPORTS        ,BLOCK DATA   ,ZONE2:TOO

// MD CMD_4 & CMD_5 shows how to omit optional information to store other than DA_Val all data array are optional
// here these mds will store only value an name of two parameters DIN2 and AV2 from zone II type block, under block data option of
//REPORTS mneu
// and will store values in DA_Val array at offsets 0, 1
//                types     DA_TYP                 0, 1
Map_Descriptors
Map_Descriptor_Name ,scan_interval ,Data_Array_Name ,Data_Array_Offset ,Da_Byte_Name ,Function  ,Node_Name  ,Gcm_Main_Menu
,Gcm_Sub_Menu ,Block_Name  ,Variable_Name ,Timeout
CMD_4               ,0.5s          ,DA_Val3         ,0                 ,DA_Name3     ,Rdbc      ,GCM_01     ,REPORTS        ,BLOCK
DATA   ,ZONE2:TOO  ,DIN2          ,5s
CMD_5               ,0.5s          ,DA_Val3         ,1                 ,DA_Name3     ,Rdbc      ,GCM_01     ,REPORTS        ,BLOCK
DATA   ,ZONE2:TOO  ,AV2           ,5s
```

4.6.2 CSV Example 2 – Read Current Status Report from a LCM

The csv file provided below has been configured to read Current Status report form LCM connected to GCM…..

```
//=========================================================================
//
//  1.00aA 30 May 03 SSS  Created
//
//=========================================================================*/


//=========================================================================
//
// Notes :  None.
//
//
//=========================================================================


//=========================================================================
//
//    Common Information
//
Bridge
Title
Gcm_Asc Testing Client


//=========================================================================
//
//    Data Arrays
//
// DA_Val---------> will contain the value of the parameter
// DA_Name--------> will contain the name of the parameter (each will occupy at least 22 bytes )
// DA_TYP---------> will contain the type parameter
// DA_Units-------> will contain the units for  value (each will occupy at least 16 bytes )

// NOTE: if "value" is not available 99 is considered as default value to show some progress during test
//       if "typ"   is not available 99 is considered as default typ to show some progress   during test
//      if "units" are not available "N/A" will stored, same will be with Digital values

// DA_TYP may have value one of the following if we found successfully
// Check with manual for most updated values
// AI                  1   // simple analog inputs
// AO                  2   // simple analog outputs
// DI                  3   //Digital Inputs
```

```
// DO                  4   // Digital outputs
// UI_A                5   // universal Input Analog
// UI_D                6   // universal Input Digital
// WINDOW_AV           7   // Window Analog  value(for micro zone II) READ ONLY
// WINDOW_DV           8   // Window Digital value(for micro zone II) READ ONLY
// EMS_AI              9   // Ems Analog  Value (for microzone II) Writable
// EMS_DI              10  // Ems Digital Value (for microzone II) Writable
// LOOP                11  // for LCM Current status


Data_Arrays
Data_Array_Name ,Data_Format ,Data_Array_Length
// These data arrays are used to hold information for at most 16 parameters
// for LCMD_1 mapdesc
DA_Val          ,float       ,16
DA_Name         ,Byte        ,352
DA_TYP          ,Byte        ,16
DA_Units        ,byte        ,256
// These data arrays are used to hold information for at most two parameters
// for LCMD_2 and LCMD_3 mapdesc
DA_Val2          ,float       ,2
DA_Name2         ,Byte        ,44
DA_TYP2          ,Byte        ,2
DA_Units2        ,byte        ,32
// In this data array you will push value and with that value one parameter will be overridden at LCM
// for LCMD_4
DA_Val3         ,float       ,1


//=============================================================================
//
//    client Side Connections
//

Connections
Port ,Baud ,Protocol ,Timeout
P1   ,9600 ,Gcm_Asc  ,5s


//=============================================================================
//
//    client Side Nodes
//

Nodes
Node_Name ,Node_ID ,Protocol ,Port ,User_name ,Password1 ,dev_type
GCM_01    ,1       ,Gcm_Asc  ,P1   ,user      ,pass      ,GCM
```

```
//===============================================================================
//
//    Client Side Map Descriptors
//
//reading maps
//Md CLCM_1 will read whole Current Status report from LCM named as LCM 2:OCC AH20 which is connected to GCM
// and will store values in DA_Val data array at 0, 1, 2, --- onward as float values
//                typ    in DA_Typ              0,1 ,2 ,----        as Integral values
//                Names  in DA_Name             0, 22,44,---- onward   (22 characters each)
//                Units in DA_Units             0, 16, 32 --- onward    (16 characters each)
Map_Descriptors
Map_Descriptor_Name ,scan_interval ,Data_Array_Name ,Data_Array_Offset ,Da_Byte_Name ,Da_Float_Name ,Da_Bit_Name  ,Function
,Node_Name  ,Gcm_Main_Menu  ,Gcm_Sub_Menu  ,LCM_Name         ,User_name ,Password1 ,Lcm_Main_Menu  ,Length
CLCM_1              ,0.5s          ,DA_Val          ,0                ,DA_NAME      ,DA_TYP        ,DA_Units     ,Rdbc      ,GCM_01
,ACCESS         ,LOCAL LCM     ,LCM  2:OCC AH20  ,user      ,pass      ,Current Status  ,1



//Md CLCM_2 & CLCM_3 will read AO:CWVLV:AV and LOOP:HTGCLG:AV parameters from Current Status report from LCM named as LCM 2:OCC AH20
//which is connected to GCM
// and will store values in DA_Val data array at 0,  1    respectively as float values
//                typ    in DA_Typ              0,  1                as Integral values
//                Names  in DA_Name             0,  22              (22 characters each)
//                Units in DA_Units             0,  16              (16 characters each)

Map_Descriptors
Map_Descriptor_Name ,scan_interval ,Data_Array_Name ,Data_Array_Offset ,Da_Byte_Name ,Da_Float_Name ,Da_Bit_Name  ,Function
,Node_Name  ,Gcm_Main_Menu  ,Gcm_Sub_Menu  ,LCM_Name         ,User_name ,Password1 ,Lcm_Main_Menu  ,Variable_name      ,Length
CLCM_2              ,0.5s          ,DA_Val2         ,0                ,DA_NAME2     ,DA_TYP2       ,DA_Units2    ,Rdbc      ,GCM_01
,ACCESS         ,LOCAL LCM     ,LCM  2:OCC AH20  ,user      ,pass     ,Current Status  ,AO:CWVLV:AV        ,1
CLCM_3              ,0.5s          ,DA_Val2         ,1                ,DA_NAME2     ,DA_TYP2       ,DA_Units2    ,Rdbc      ,GCM_01
,ACCESS         ,LOCAL LCM     ,LCM  2:OCC AH20  ,user      ,pass     ,Current Status  ,LOOP:HTGCLG:AV     ,1



//Md CLCM_4 will override the parameter AO:SSS2 with a new value taking from offset 0 at DA_val3 data array
//writing maps
Map_Descriptors
Map_Descriptor_Name ,scan_interval ,Data_Array_Name ,Data_Array_Offset ,Function  ,Node_Name  ,Gcm_Main_Menu  ,Gcm_Sub_Menu ,LCM_Name
,User_name ,Password1 ,Lcm_Main_Menu  ,LCM_Sub_Menu ,Variable_Name ,Length
CLCM_4              ,0.5s          ,DA_Val3         ,0                ,wrbc       ,GCM_01     ,ACCESS         ,LOCAL LCM   ,LCM
2:OCC AH20  ,user      ,pass      ,BLOCKS         ,OVERRIDE     ,AO:SSS2       ,1
```

5 Storage Information

5.1 Variable Value

If the value has been found it will be stored as a float value.
If variable represents states ON/OFF then 1.0/0.0 will be stored.
If variable found but value doesn't, then 99.0 will be stored instead and this value should be considered the indication that value is not available because of some communication error between GCM and other connected devices.

5.2 Variable Names

Internally driver reserved 22 characters for storage of name as this is the biggest name known at time of development.

5.3 Variable Types

Variables types describe the nature of variables like Analog Input, Digital Output etc.
But driver will store numeric number corresponding to these types . See the table below :

Variable Type	Numeric Value
Analog Input (AI)	1
Analog Output (AO)	2
Digital Input (DI)	3
Digital Output (DO)	4
UI Analog (UIA)	5
UI Digital (UID)	6
Window Analog Value (WINDOWAV)	7
Window Digital Value (WINDOWDV)	8
EMS Analog Value (EMSAV)	9
EMS Digital Value (EMSDV)	10
LOOP	11

5.4 Variable Units

If the value has been found its units will be stored in data array to hold units .Driver internally reserved 16 characters to store units.
If value will just state ON/OFF or if value not found then “N/A” will be stored instead of units indicating that either units are not applicable for state values or units are not available because value is also not available.

If value extracted from the response will be stored in the array called DA_Val.
Variable name will be stored in data array DA_Name.
Variable type will be stored in data array DA_Typ.
Variable units will be stored in data array DA_Units.
Ensure that it is suitable format for storing value – FLOAT is suggested.
Driver will store information in the following manar.
For first variable
Variable value will be stored at 0th offset at Da_Val data array.
Variable name will be stored at 0th offset through 21st offset at Da_Name data array.
Variable type will be stored at 0th offset at Da_typ data array
Variable units will be stored at 0th offset through 15th offset at Da_Units data array.

For Second variable
Variable value will be stored at 1th offset at Da_Val data array.
Variable name will be stored at 22nd offset through 43 rd offset at Da_Name data array.
Variable type will be stored at 1th offset at Da_typ data array
Variable units will be stored at 16th offset through 31th offset at Da_Units data array.
And so on…..

6 Driver Notes

6.1 Driver Limitations and Exclusions

As a client:

The driver can **Only** read reports for block data of type ZONE2 and Current Status for LCM (model no.LCM-88210-1) and it can read only variable name, value, units and type only. This driver is tested override (write) block's variable for specified LCM.

The driver cannot be configured to act as server.

The driver performance is slow. In situations where the driver is configured to read data from multiple remote devices or multiple device types expect response times that may range up to 10 seconds. For example: If the driver has been configured to read data from more than one LCM connected to the GCM then performance will be slow because the driver needs to log into each LCM, navigate the menus, log out and log into the next LCM.

During testing variations in GCM, LCM and other device firmware were noted. Of the three GCM units used for testing, all three produced slight variations in the message stream. It is possible that other versions of firmware may produce other variations which may cause problems for the driver.

As a server:

Driver can not be configured as Server.

6.2 Driver Error Messages

The driver reports information and errors to you in the form of messages printed to the error log. Those messages marked with a * are only printed once even if they occur repeatedly.

GCM_Asc:#1 Err. user name %s is not valid
The user name is either blank or of more than 16 characters.

GCM_Asc:#2 Err. password %s is not valid.
The password is either blank or of more than 16 characters.

GCM_Asc:#3 Err. dev_type %s is not valid
The dev type is either blank or more thank 16 characters.

GCM_Asc:#4 Err. Illegal Map Descriptor's GCM_MAIN_MENU parameter
GCM main menu option is not specified.

GCM_Asc:#5 Err. Illegal Map Descriptor's GCM_SUB_MENU parameter
GCM main menu option is not specified.

GCM_Asc:#6 FYI. Undefined Dev_Type, Using <%s> by default
You did not defined the dev type while declaring nodes driver proceeding with default dev type.

GCM_ASC:#7 data array %s is short to store
This data array declared to store values is insufficient to store all values , increase the length parameter of this data array.

GCM_ASC:#8 data array %s is short to store
This data array declared to store variable names is insufficient to store all values , increase the length parameter of this data array.

GCM_ASC:#9 data array %s is short to store
This data array declared to store variable types is insufficient to store all values , increase the length parameter of this data array.

GCM_ASC:#10 data array %s is short to store
This data array declared to store variable types is insufficient to store all values , increase the length parameter of this data array.

7 Connecting through Hyper Terminal

To determine the values of

- GCM_Main_Menu
- GCM_Sub__Menu
- LCM_Name
- Lcm_Main_Menu
- LCM_Sub_Menu
- Variable_Name
- Block_Name
-

Connect to the GCM through hyper terminal

7.1 Create a hyper Terminal connection

a)Click on start, programs , accessories, communications, Hyper terminal
Give some name to this connection going be build
Like “GCMconnection”
b)Press OK
On the next window you will see or chosse COM1 or COM2
c)Press OK
Now you will see some port settings
d)Adjust the setting like
Bits per second 9600
Parity None
Stop bits 1
Flow control None

e)Press OK

Now your PC is connected to GCM through Hyper Terminal

Closing Hyper Terminal Connection
when you done with hyper terminal just click at cross button of window to close
it will ask you to “are sure to disconnect ? “
Press on YES
Now it will ask you to “save the current session ?”
Press YES
Now your “GCMconnection” will saved next when you will connect through hyper terminal you can simply open this (GCMconnection) to connect, no need to adjust the settings again.

7.2 Browsing GCM through hyper terminal

7.3 Determine all parameter values required to read Current status for Lcm

To know all variables required to read Current Status or override at Lcm
Connect your PC Comport to GCM at comm. Port 2 taking care about
Pin 2(TX) at GCM should connect to pin 3(RX) at PC

Pin 3(RX) at GCM should connect to pin 2(TX) at PC
Pin 7(Ref) at GCM should connect to pin 5(Ref) at PC
When ever you see one of the following message on screen press escape (Esc) and try again checking your spelling, spaces between characters or so

ACCESS NOT ALLOWED
INVALID ACCESS CODE
WHAT?
NO SELECTIONS
State1. When you will first connect to GCM it may be in any state now should reboot the GCM
And wait for at least 20 second during which GCM will to do some initial tests.
And you can see

SIEBE E.C.
Barber-Colman
Network-8000
Dev Type: GCM
Ver: ASD
Rev: 5.4A
Eng Rev: 5.4A
Date: 4/14/99
Port: PORT 2
EPROM test PASSED
SYSRAM test PASSED
BLKRAM test PASSED
CLOCK test PASSED

Here you can clearly see Dev Type:GCM so GCM will be value of **dev_type** parameter used while declaring nodes.

In some models when we press “enter” GCM ask for
USER NAME:
For other models we have press “enter” 3-4 times with some time gap.
State2:
When you see
USER NAME:
Type user name and this will be the value for **User_name** parameter used while declaring nodes and hit enter, now you will see

PASSWORD:
Type password and this will be the value for **password1** parameter used while declaring nodes and hit enter,
State3:
now you will see

GCM 1:BARBER-COLMAN

MAIN MENU
CURRENT STATUS

Now you are main menu screen of GCM. Press right arrow / left arrow key to see other options. If you are unable to see other options then try again after downloading lastest version of Hyper Terminal.

Here we have to select one option that will be the value for the parameter GCM_Main_Menu.

State4:

a)If want to read Lcm press the right/left arrow key too see

GCM 1:BARBER-COLMAN

MAIN MENU
ACCESS

Here "ACCESS" is value for our **GCM_Main_Menu** parameter.

State5:
b)If want to read Micro Zone II device press the right/left arrow key too see

GCM 1:BARBER-COLMAN

MAIN MENU
REPORTS

Here "REPORTS" is value for our **GCM_Main_Menu** parameter.

Now press "enter" after choosing your main menu option (ACCESS is considered here)

State6:
Now you will see

GCM 1:BARBER-COLMAN

ACCESS MENU
LOCAL LCMs

Here "LOCAL LCM" is value for our **GCM_Sub_Menu** parameter.

Now again hit "enter"
State7:
you will see

GCM 1:BARBER-COLMAN

SELECT LCM
LCM 2:OCC AH20

Here "LCM 2:OCC AH20" will be the value for our **Lcm_Name** parameter if this is only Lcm connected to GCM otherwise choose other Lcm name by pressing right/left arrow key.
After selecting required Lcm press "enter"
State8:
you will be within few seconds

LCM 2:OCC AH20

USER NAME:
USER

It is found that mostly the user name and password for Lcm remains the same as for GCM. In this case USER was the user name for GCM and Lcm.
So “USER” is a value for **User_name** . (the one will be used in Map descriptor)

Now just press “enter” or you can type user name and then pres enter
next you will see

```
LCM  2:OCC AH20

PASSWORD:
******
```

Now just press “enter” or you can type password to continue but this password will be value for the parameter **Password1** used in map descriptors
State9:
Now you will see

```
LCM  2:OCC AH20

MAIN MENU
CURRENT STATUS
```

Now you are at main menu screen of Lcm and you can see the other options by pressing the right/left arrow keys.
a)if you want to read current status then CURRENT STATUS will be the value for **Lcm_Main_Menu** parameter.
b) If you want override in Lcm then press the right/left arrow and see the following
State10:
Now you will see

```
LCM  2:OCC AH20

MAIN MENU
BLOCKS
```

Here BLOCKS will the value of **Lcm_Main_Menu** parameter.

Now to read current status select current status option and hit enter.
Now you will see
State11:
you will see the following, it may take some time to paint full screen

```
 NET01:GCM01:LCM02   Ver 5.4   OCC AH20          3/12/94  3:15:48

AI:STATIC:AV              NOT ACTIVE OUTPUT
LOOP:CONE:CALSP           ABNORMAL NO INPUT
AI:SAT:AV                 0.00 VOLTS
LOOP:HTGCLG:CALSP         70.0 DEG F
AI:RAT:AV                 ABNORMAL INPUT
LOOP:RMTSP:CALSP          ABNORMAL NO INPUT
DO:SFSS:DV                OFF
DI:SFRUN:DV               OFF
AO:MADMPR:AV              1.00 VOLTS
AO:HWVLV:AV               5.50 VOLTS
AO:CWVLV:AV               9.50 VOLTS
LOOP:HTGCLG:AV            0.0 %
AO:SFCONE:AV              CANT MATCH ATTRIB
LOOP:CONE:AV              ABNORMAL NO INPUT
AO:SFCONE:CAUSE           CANT MATCH ATTRIB
DI:SFFLTR:DV              CANT MATCH ATTRIB
```

ACCES NET01:GCM01:LCM02 3/12/94 3:03 USER LOGGED ON
DIAGB NET01:GCM01:LCM02 9/15/03 15:42 AI:RAT SENSOR RANGE

This is Current Status Screen for Lcm. Here all the lines like
AO:HWVLV:AV 5.50 VOLTS
are considered process able lines.
Here AO:HWVLV:AV is considered as **Variable_name i**f you want to read single parameter otherwise skip variable_name parameter to read all variables. But count all the lines for example in this case it is 16 and data array length must be specify to accommodate all these variables.
Eg. Length for Da_val will be be 16
Length for Da_name will be 22*16 = 352
Length for Da-typ will be 16
Length for Da_units will be 16*16 =256

This was the final state for reading Lcm Current Status and got all the required information for reading as a whole or a single variable.

7.4 Determine all parameter values required to override variables for Lcm

You can restart the GCM and follow the same procedure as above and note down parameters as you follow the procedure to come at State10.
Till now we discovered values for following parameters
User_name
Password1
GCM_main_menu
GCM_Sub_menu
Lcm_name
Lcm_main_menu

Now at state10 again press right/left arrow keys to see

LCM 2:OCC AH20

MAIN MENU
BLOCKS

Here BLOCKS is a value for the parameter **Lcm_Main_Menu.**
Now press enter and you will see

NET01:GCM01:LCM02 Ver 5.4 OCC AH20 3/12/94 3:28:38
LCM 2:OCC AH20

BLOCKS MENU
OVERRIDE

Here OVERRIDE is a value for the parameter **Lcm_Sub_Menu.**
Now again press enter
You will see

LCM 2:OCC AH20

SEL BLOCK TO OVERRIDE
AI:RARH

You can scroll to see all variables (blocks) that can be override but driver has been tested with AO type variables like

NET01:GCM01:LCM02 Ver 5.4 OCC AH20 3/12/94 3:28:38
LCM 2:OCC AH20

SEL BLOCK TO OVERRIDE
AO:SSS2

Here AO:SSS2 is the value of **Variable_Name** parameter.
Now we have all required information to override variables in Lcm.

7.5 Determine all parameter values required to read variables for micro Zone II

Restart the GCM and follow the procedure to come at state6.
At this point we values for the following parameters

User_name,
Password1,
GCM_main_menu

Only Block_name and Variable_ parameter is remaining to be known.

Now at state6 press enter
You will see

GCM 1:BARBER-COLMAN

REPORTS MENU
ACTIVE ALARMS

Now you can press right/left arrow keys to see more options under Reports Menu
And come to see

GCM 1:BARBER-COLMAN

REPORTS MENU
BLOCK DATA

Here BLOCK DATA is the value of **GCM_sub_menu**.
Now press enter and see similar to the following

NET01:GCM01:LCM02 Ver 5.4 OCC AH20 3/12/94 3:28:38
GCM 1:BARBER-COLMAN

SEL BLOCK TO REPORT
AO:PMC1

Here you can press right/left arrow keys to see other blocks for which you can get a report, but driver is implemented to read only ZONE2 type blocks those contain data for Micro zone II device, So scroll and come to see similar to

GCM 1:BARBER-COLMAN

SEL BLOCK TO REPORT
ZONE2:TOO

Here ZONE2:TOO is the value of **Block_Name** parameter.
Now you have all required information for describing a map desc to read all variables for Microzone device

Now press enter to see

GCM 1:BARBER-COLMAN

SEND REPORT
TO PRINT GROUP

Now press right/left arrow key to see

GCM 1:BARBER-COLMAN

SEND REPORT
TO LOCAL TERMINAL

Now press enter to see

BLOCK DATA
NET01:GCM01:LOCAL Ver 5.4GL BARBER-COLMAN 8/19/93 6:12pm

ZONE2:TOO

```
UPTIM  UPDATE TIME          5 SEC
ADDR   PHYSICAL ADDRESS     1
TYUI1  INPUT TYPE UI1       ANALOG
UNUI1  UNITS FOR UI1        DEG F ** .1
TYUI2  INPUT TYPE UI2       ANALOG
UNUI2  UNITS FOR UI2        DEG F ** .1
TYUI3  INPUT TYPE UI3       DIGITAL
TYUI4  INPUT TYPE UI4       ANALOG
UNUI4  UNITS FOR UI4        DEG F ** .1
TYUI5  INPUT TYPE UI5       ANALOG
UNUI5  UNITS FOR UI5        DEG F ** .1
TYUI6  INPUT TYPE UI6       ANALOG
UNUI6  UNITS FOR UI6        DEG F ** .1
TYUI7  INPUT TYPE UI7       ANALOG
UNUI7  UNITS FOR UI7        DEG F ** .1
TYUI8  INPUT TYPE UI8       ANALOG
UNUI8  UNITS FOR UI8        DEG F ** .1
NUMDI  NUM OF EMS DIGITAL   4
NUMAI  NUM OF EMS ANALOG    4
NUMDV  NUM OF WIN DIGITAL   4
NUMAV  NUM OF WIN ANALOG    4
AV1    UNITS FOR WINDOW AV1 DEG F ** .1
UAV2   UNITS FOR WINDOW AV2 DEG F ** .1
UAV3   UNITS FOR WINDOW AV3 DEG F ** .1
UAV4   UNITS FOR WINDOW AV4 DEG F ** .1
EXENA  EXCEPTION ENABLE     0
CENAB  CALCULATE ENABLE     ON
EMSCO  EMS CONTROL          ON
DIN1   EMS DV1              EMPTY
DIN2   EMS DV2              EMPTY
DIN3   EMS DV3              EMPTY
DIN4   EMS DV4              EMPTY
AIN1   EMS AV1              EMPTY
AIN2   EMS AV2              EMPTY
AIN3   EMS AV3              EMPTY
AIN4   EMS AV4              EMPTY
UIAV1  UI ANALOG VALUE 1    249.6 DEG F
UIAV2  UI ANALOG VALUE 2    0.0 DEG F
UIDV3  UI DIGITAL VALUE 3   OFF
UIAV4  UI ANALOG VALUE 4    0.0 DEG F
UIAV5  UI ANALOG VALUE 5    0.0 DEG F
UIAV6  UI ANALOG VALUE 6    0.0 DEG F
UIAV7  UI ANALOG VALUE 7    0.0 DEG F
UIAV8  UI ANALOG VALUE 8    0.0 DEG F
DO1    DIGITAL OUTPUT 1     OFF
DO2    DIGITAL OUTPUT 2     OFF
DO3    DIGITAL OUTPUT 3     OFF
DO4    DIGITAL OUTPUT 4     OFF
```

```
DO5   DIGITAL OUTPUT 5     OFF
DO6   DIGITAL OUTPUT 6     OFF
DO7   DIGITAL OUTPUT 7     OFF
DO8   DIGITAL OUTPUT 8     OFF
AO1   ANALOG OUTPUT 1       0.0 %
AO2   ANALOG OUTPUT 2       0.0 %
AO3   ANALOG OUTPUT 3       0.0 %
AO4   ANALOG OUTPUT 4       0.0 %
DV1   WINDOW DV1          OFF
DV2   WINDOW DV2          OFF
DV3   WINDOW DV3          OFF
DV4   WINDOW DV4          OFF
AV1   WINDOW AV1          0.0 DEG F
AV2   WINDOW AV2          0.0 DEG F
AV3   WINDOW AV3          0.0 DEG F
AV4   WINDOW AV4          0.0 DEG F

RESET: 00  FREEZE: 00  LAST EXEC:  6:12:31pm  NEXT EXEC:  6:12:37pm
PRESS <ESC> TO CONT
```

This is the report for microzone II device. All the lines starting with
DIN
AIN
UIA
UID
DO
AO
DV
AV
are considered to be process able those contains some information like
UIAV1 UI ANALOG VALUE 1   249.6 DEG F
Line will be parsed as
UIAVI as **Variable_Name** if you want to read only this variable and also this will be the variable name that will be stored in Da_Name data array
UIA here actually is the type of of the variable (Universal Input Analog)
249.6 is the value of this variable.
DEG F are the units for this variable.
Now we have all required information for poll GCM for Zone2 type blocks to read as a whole then skip the Variable_name parameter and to read a single variable mention the Variabl_name we just found.
Here once again there are 36 variable in total for this report so if you are going to read this whole block you data array length must as follows
Eg. Length for Da_val will be be 36
Length for Da_name will be 22*36 = 792
Length for Da-typ will be 16
Length for Da_units will be 16*36 =576

8 Revision History

Date	Driver Version	Document Revision	Comment
16Sep2003	1.00	0	Draft release by SSS.
03Oct2003		1	Reviewed with minor corrections by PMC Added CSV file examples in section 4.6