



User's Manual

MFC-PROFIBUS Interface





QUALIFLOW S.A.

Le Millénaire
350, rue Alfred Nobel – B.P. 7
34935 MONTPELLIER - CEDEX 9
France
Ph#: +33 (0)4 67 99 47 47
Fax#: +33 (0)4 67 99 47 48

QUALIFLOW Technology Center

44862 Osgood Road
CA-94539
Fremont
CALIFORNIA – USA
Ph#: +1 (510) 440 93 74
Fax#: +1 (510) 440 93 75

QUALIFLOW Japan

3F Hattori Bldg.
30 Yotsuya
4 – Chome Shijuku-ku
Tokyo 160-0004
Japan
Ph#: +81 (0)3 5368 1682
Fax#: +81 (0)3 5368 1683

Identification					
Reference	D 10-088	Revision	V00	Date	13 Feb 2003
Document name	MFC-PROFIBUS Interface User's Manual				
File name	MFC_ProfibusInterface_UserManual_V00.doc				

Control					
Author		Verified		Approved	
Visa	Date	Visa	Date	Visa	Date
Gilles Sanchez	13 Feb 2003	Albin Diranzo	05 Mar 2003	Pascale Garnier	07 Mar 2003

History				
Author	Date	Description	Revision	Status
Gilles Sanchez	15 Feb 2003	Initial version	00	

TABLE of CONTENTS

1.	Description.....	1
2.	PROFIBUS overview	1
2.1.	SCOPE.....	1
2.2.	REGISTRATION	1
2.3.	REFERENCES	1
2.4.	PART NUMBERS	1
3.	PROFIBUS interface installation	2
3.1.	PRINCIPLE	2
3.2.	DIMENSIONS.....	3
3.3.	CONNECTIONS.....	4
4.	Features.....	5
4.1.	ELECTRICAL FEATURES	5
4.2.	PROFIBUS FEATURES	5
5.	Modules	6
6.	Classes	7
6.1.	CLASS 1 (DATA MINI):.....	7
6.2.	CLASS 2 (DATA EXTENSION):	7
6.3.	CLASS 3 (CONFIGURATION):	7
6.4.	CLASS 4 (IDENTITY):.....	7
7.	Data format	8
8.	Data layout and offsets	9
8.1.	INTRODUCTION	9
8.2.	CLASS 1.....	9
8.3.	CLASS 1 AND 2	9
8.4.	CLASS 1, 2 AND 3	10
8.5.	CLASS 1, 2, 3 AND 4	10
9.	Appendix 1: GSD file.....	11
10.	Appendix 2 - Definitions	13

1. Description

This document describes the module interface Qualiflow uses to integrate MFCs into a Profibus network. This interface is based upon a standard "off the shelf" product manufactured by a third company specialized in Profibus. This product integrates a micro-controller as well as an ASIC to insure a proper Profibus protocol. A Qualiflow' specific firmware loaded into the flash memory guarantees a proper **Profibus to Modbus** translation.

The device, when loaded with Qualiflow's firmware, is referred to as MFC-Profibus Interface.

2. PROFIBUS overview

2.1. Scope

The **MFC-Profibus Interface** is dedicated to the use of Qualiflow's Digital MFCs. It can interface up to four Digital MFCs.

2.2. Registration

MFC-Profibus Interface has been registered on 18 January 2001 by PROFIBUS Nutzer organization e.V. with the following details:

- | | |
|------------------|-------------------------------|
| • Model name: | MFC-PROFIBUS INTERFACE |
| • Release: | 1.0 |
| • Manufacturer: | QUALIFLOW S.A. |
| • Device type: | Slave |
| • Protocol type: | DP |
| • ID number: | 05B6 |
| • GSD file: | QUAL05B6.GSD |

2.3. References

The following documents were used as basis for constructing this present document. They represent also an interesting source of information.

- Ref 1. [CENELEC PROFI] CENELEC EN50170-2 Profibus 1996
- Ref 2. [MODBUS] Modicon Modbus Protocol Reference Guide: PI-MBUS-300 Rev. J
- Ref 3. [IEEE-754] Standard for Binary Floating-Point Arithmetic
- Ref 4. [SEMI E52-1000] Practice for referencing gases used in digital mass flow controllers

2.4. Part numbers

- | | |
|--|--------------------|
| • MFC-Profibus Interface: | Q2001733-25 |
| • RS-485 Cable MFC ↔ MFC: | Q2001733-18 |
| • RS-485 Cable MFC-Profibus Interface ↔ MFC: | Q2001733-26 |

3. PROFIBUS interface installation

3.1. Principle

Digital MFCs communicate with host computers using a RS-485 network and Modbus protocol (see **Ref 2**). In order to connect the Digital MFCs to a Profibus network, the MFC-Profibus Interface is used.

MFC-Profibus Interface connects to Profibus network by one side and to RS-485 Modbus by the other side. Two sixteen rotary switches are provided for configuring the address of the device.

The MFC-Profibus Interface operates as a **Profibus slave** and **Modbus master**. Up to four MFCs can be connected to one MFC-Profibus Interface. MFCs operate as Modbus slaves and must be configured to respond addresses 1 to 4. This is performed using the software Digisoft, available from Qualiflow's web site.

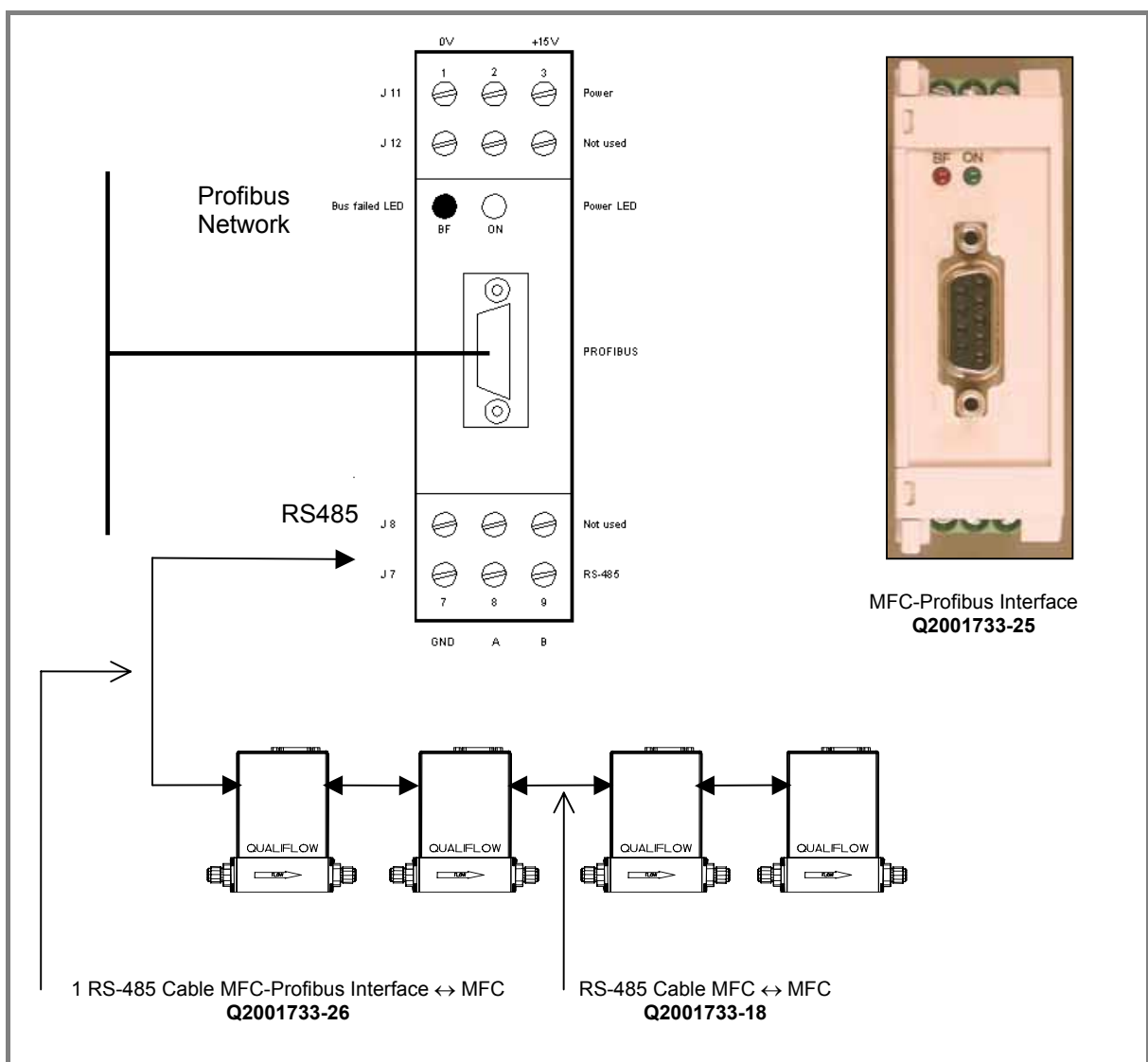


Figure 1: Principle

3.2. Dimensions

The figure below shows the implantation of the Profibus interface and main dimensions in millimeters. The device fits on a standard **Omega rail**.

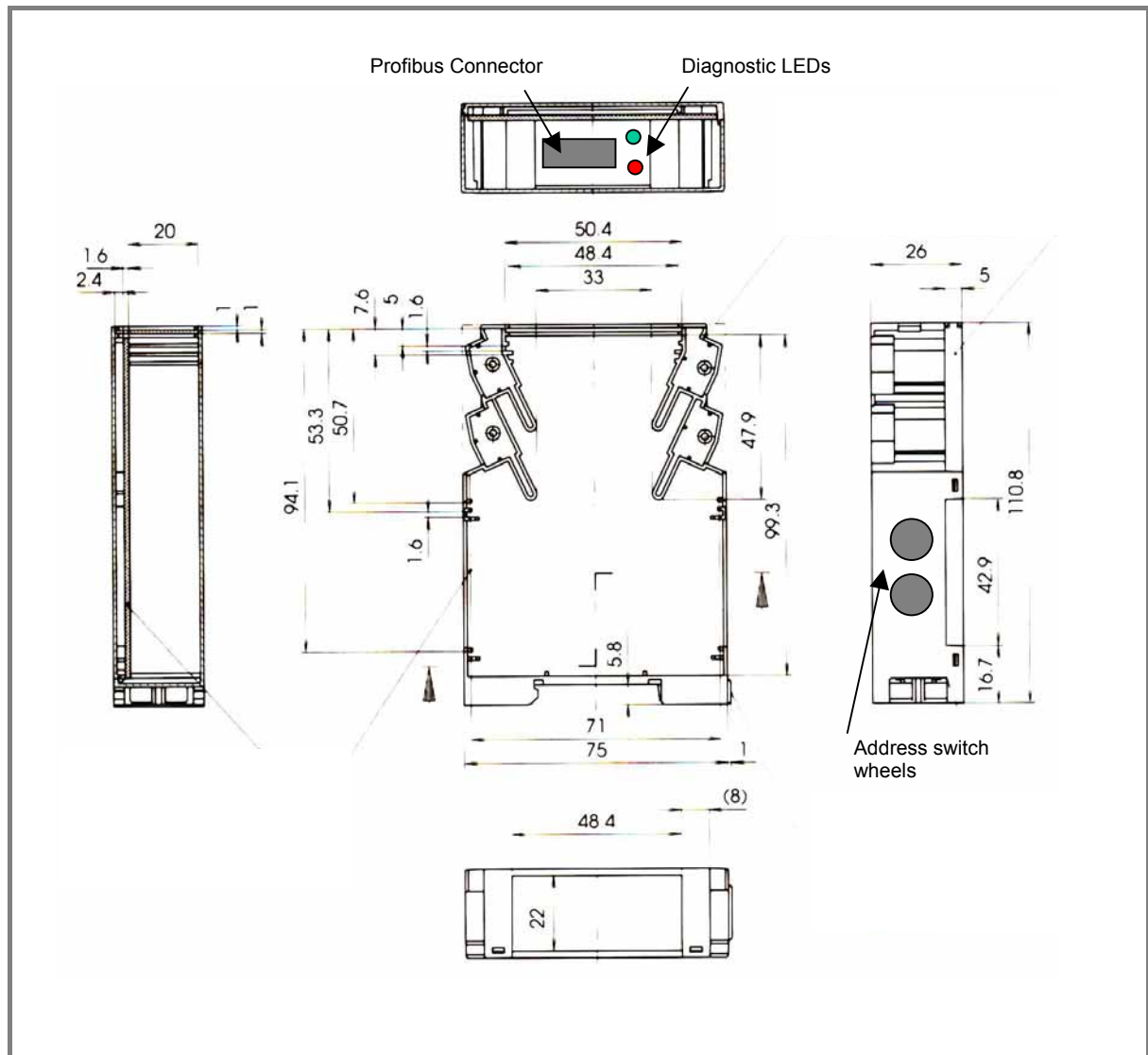


Figure 2: Dimensions

3.3. Connections

The device front face is as follows:

- The top part of the interface includes power supply connections;
- The bottom part receives the RS485 connection coming from the first digital MFC;
- Two leds show the device status when powered on;
- A 16-pin plug at the center allows connecting the PROFIBUS network

RS485 connection description:

- Cable PN: **2990986Q**
- Wire: **Green to 8(A)**
- Wire: **Red to 9(B)**

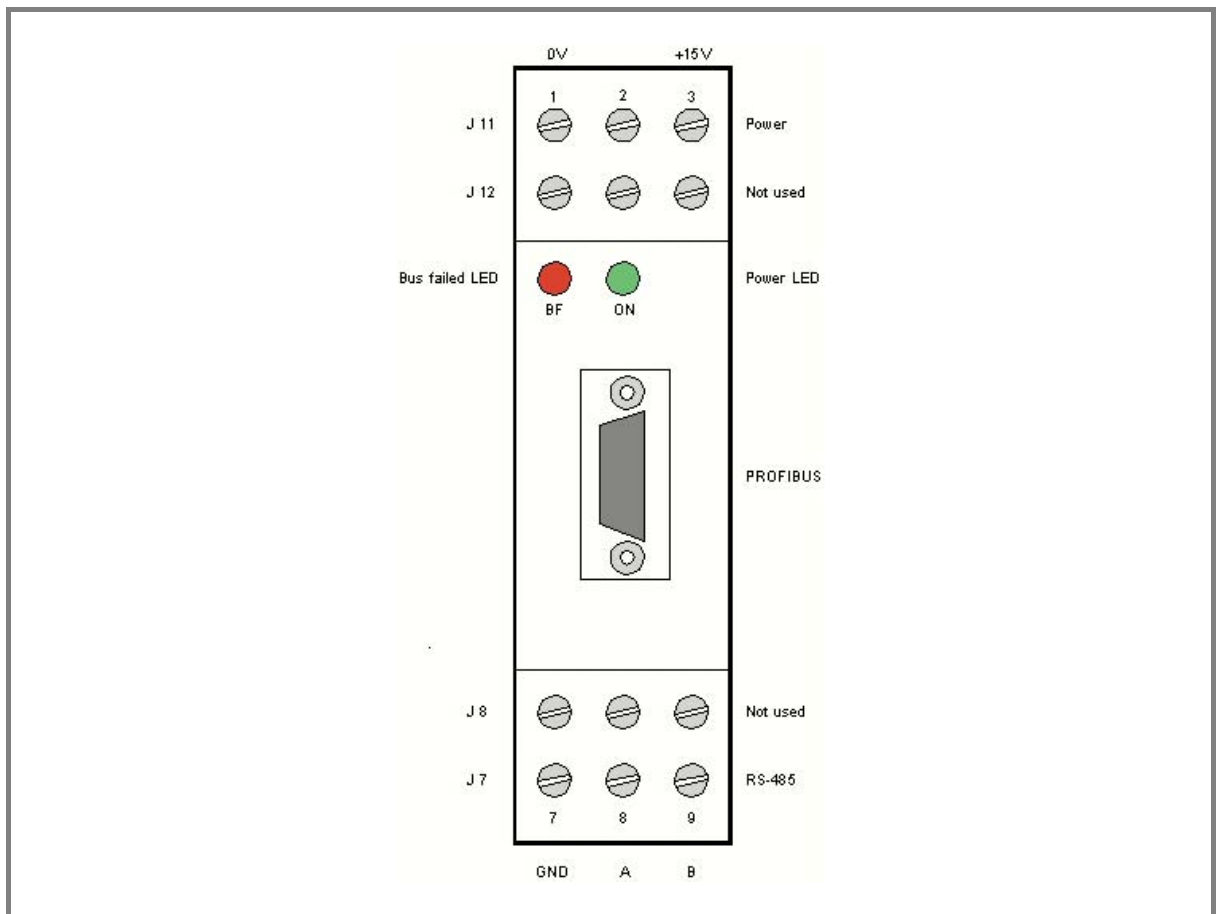


Figure 3: Connections

4. Features

4.1. Electrical features

PROFIBUS Interface	DP slave 9,6K to 12M bauds
PROFIBUS Connection	Sub D 9 pins
MODBUS Connection	Terminal block
Power supply connection	Terminal block
PROFIBUS Address	2 switch wheels
MODBUS Address	Software defined
Diagnostic	LED bus fail; LED power ON
Consumption	2.6 W
Alimentation	10 to 30V CC
Dimension	See Figure 2: Dimensions

Table 1 : Electrical specifications

4.2. Profibus features

Bus type	PROFIBUS-DP EN50170
Protocol stack	SIEMENS
Automatic speed detection	9.6 Kbit – 12 Mbit
Media	RS-485
Connector	SUBD 9 fem.
Insulation	A and B : optical
Profibus-DP communication	SIEMENS SPC3
LED	Bus failed
Address range	1 to 126 with switch wheels
Data transmission	Cyclic data transmission. DP-V1 is not supported.

Table 2 : Profibus specifications

5. Modules

The Profibus interface is a modular interface. It is possible to choose the data we want to transfer to or from the MFC. To do so, 16 modules are available.

A module sets the configuration of the PROFIBUS interface. Choosing a module means setting up the number of digital MFCs and the variables communicated between the MFC and the PROFIBUS network.

Each module is organized in classes as shown bellow:

• Module 1	1 MFC datamin
• Module 2	2 MFC datamin
• Module 3	3 MFC datamin
• Module 4	4 MFC datamin
• Module 5	1 MFC datamin + dataext
• Module 6	2 MFC datamin + dataext
• Module 7	3 MFC datamin + dataext
• Module 8	4 MFC datamin + dataext
• Module 9	1 MFC datamin + dataext + config
• Module 10	2 MFC datamin + dataext + config
• Module 11	4 MFC datamin + dataext + config
• Module 12	3 MFC datamin + dataext + config
• Module 13	1 MFC datamin + dataext + config + id
• Module 14	2 MFC datamin + dataext + config + id
• Module 15	3 MFC datamin + dataext + config + id
• Module 16	4 MFC datamin + dataext + config + id

In this list, "**datamin**", "**dataext**", "**config**" and "**id**" are the different **classes** available. Refer to **section 6** for further details about **Classes**.

• Datamin	Class 1
• Dataext	Class 2
• Config	Class 3
• Id	Class 4

Each class includes a group of variables.

Example:

If you set your interface to **Module 14** configuration, you will work with **two digital MFC** and use **Class 1 + Class 2 + Class 3 + Class 4** parameters for **each MFC**.

See **section 6** for further details about **MFC variables**.

6. Classes

The different following classes set the variables available for a digital Qualiflow' MFC control board. Refer to the appropriate **MFC user's manual** for further information.

A **Class** defines a set of variables. They form an ordered list. Each class includes variables starting from the most important parameters, included in **Class 1 (Readout and WriteSetPoint variables)** to the less used parameters, included in **Class 4** (identification parameters).

This classification allows the user to **choose its parameters** and to **limit the volume of data** transferred between the program and the MFC.

At least, **Class 1** must be used. If we refer to modules (see section 5), the minimum configuration matches module 1, module 2, module 3 and module 4.

Features of each parameter listed below are developed in **section 7** in this manual.

6.1. **Class 1 (Data mini):**

- Readout
- WriteSetPoint

6.2. **Class 2 (Data extension):**

- ReadSetpoint
- Alarm code
- ReadOverRide
- WriteOverRide
- Reset Alarm

6.3. **Class 3 (Configuration):**

- Full Scale
- Gas symbol
- WriteCalibrationSlot
- ReadCalibrationSlot

6.4. **Class 4 (Identity):**

- MFC Serial number
- Product identifier
- Manufacturer identifier
- Software version

7. Data format

The following table shows the major features of each item: the **data format** and **direction**. This table is the basis to program the application that controls the Profibus interface.

Item	Variables	Direction	Type Profibus	Refresh Rate	Format	Description
1	Readout	Slave → Master	4 bytes input	500ms	Float IEEE754 see Ref 3	Actual flow in percent of Full Scale (See Ref 3 for detailed float coding)
2	ReadSetpoint	Slave → Master	2 bytes input	500ms	Unsigned integer 16 bits	Actual setpoint in binary form : 0 → 0 % of Full Scale; 65535 → 100 % of Full Scale
3	Alarm code	Slave → Master	1 byte input	500ms	BYTE	Note : A value of 255 means that the interface can't initiate communication with the MFC.
4	ReadOverride	Slave → Master	1 byte input	500ms	BYTE	Actual Valve status: 0 - Regulation; 1 - Closed; 2 - Open.
5	Full scale	Slave → Master	4 bytes input	At startup	Float IEEE754 see Ref 3	Full scale of current calibration slot (in sccm).
6	Gas symbol	Slave → Master	6 bytes input	At startup	ASCII see Ref 4	Gas symbol of current calibration slot
7	Read Calibration slot	Slave → Master	1 byte input	At startup	BYTE	Calibration slot : 0 → 9
8	MFC Serial number	Slave → Master	10 bytes input	At startup	ASCII	MFC Serial number : Should match MFC' serial number.
9	Product identifier	Slave → Master	10 bytes input	At startup	ASCII	Product identifier : Should match MFC's product identifier.
10	Manufacturer identifier	Slave → Master	4 bytes input	At startup	ASCII	Manufacturer identifier : 'QUAL'
11	Software version	Slave → Master	4 bytes input	At startup	ASCII	Software version : In the form of "x.yy" where x is the major revision and yy is the minor revision
12	WriteSetPoint	Master → Slave	2 bytes output	On state change	Unsigned integer 16 bits	Setpoint in binary form : 0 → 0 % of Full Scale; 65535 → 100 % of Full Scale
13	WriteOverride	Master → Slave	1 byte output	On state change	BYTE	WriteOverride : see ReadOverride.
14	Clear alarm	Master → Slave	1 byte output	On state change	BYTE	Clear alarm : Alarm clears on a transition from any value to \$FF.
15	Write Calibration Slot	Master → Slave	1 byte output	On state change	BYTE	WriteCalibrationSlot : 0 → 9

Table 3 : Data format

Note : 'Refresh Rate' indicates the actual rate the data is polled from or transferred to the MFC.

8. Data layout and offsets

8.1. Introduction

The following data tables set the **position** of the variable on the frame and the **frame length** for PROFIBUS inputs / outputs depending on the **variable**. The **Offset** represents the position of the variable on the frame. The length is expressed in byte.

Example: Class 2 – Two MFCs configuration.

For **MFC 2**, the **ReadSetpoint** variable needs **2 bytes** on the frame. Its **offset** on the frame is **12**.

PROFIBUS inputs			Offset			
Class	Variable name	Length	MFC 1	MFC 2	MFC 3	MFC 4
1	Readout	4	0	8	16	24
2	Read Setpoint	2	4	12	20	28

8.2. Class 1

Frame data layout for modules including Class 1 only.

PROFIBUS inputs			Offset			
Class	Variable name	Length	MFC 1	MFC 2	MFC 3	MFC 4
1	Readout	4	0	4	8	12
TOTAL frame length			4	8	12	16

PROFIBUS outputs			Offset			
Class	Variable name	Length	MFC 1	MFC 2	MFC 3	MFC 4
1	Write Setpoint	2	0	2	4	6
TOTAL frame length			2	4	6	8

8.3. Class 1 and 2

Frame data layout for modules including Class 1 and 2.

PROFIBUS inputs			Offset			
Class	Variable name	Length	MFC 1	MFC 2	MFC 3	MFC 4
1	Readout	4	0	8	16	24
2	Read Setpoint	2	4	12	20	28
2	Alarm Code	1	6	14	22	30
2	Read Override	1	7	15	23	31
TOTAL frame length			8	16	24	32

PROFIBUS outputs			Offset			
Class	Variable name	Length	MFC 1	MFC 2	MFC 3	MFC 4
1	Write Setpoint	2	0	4	8	12
2	Write Override	1	2	6	10	14
2	Reset Alarm	1	3	7	11	15
TOTAL frame length			4	8	12	16

8.4. Class 1, 2 and 3

Frame data layout for modules including Class 1, 2 and 4.

PROFIBUS inputs			Offset			
Class	Variable name	Length	MFC 1	MFC 2	MFC 3	MFC 4
1	Readout	4	0	19	38	57
2	Read Setpoint	2	4	23	42	61
2	Alarm Code	1	6	25	44	63
2	Read Override	1	7	26	45	64
3	Full Scale	4	8	27	46	65
3	Gas Symbol	6	12	31	50	69
3	Read calibration Slot	1	18	37	56	75
TOTAL frame length			19	38	57	76

PROFIBUS outputs			Offset			
Class	Variable name	Length	MFC 1	MFC 2	MFC 3	MFC 4
1	Write Setpoint	2	0	5	10	15
2	Write Override	1	2	7	12	17
2	Reset Alarm	1	3	8	13	18
3	Write Calibration Slot	1	4	9	14	19
TOTAL frame length			5	10	15	20

8.5. Class 1, 2, 3 and 4

Frame data layout for modules including Class 1, 2, 3, and 4.

PROFIBUS inputs			Offset			
Class	Variable name	Length	MFC 1	MFC 2	MFC 3	MFC 4
1	Readout	4	0	47	94	141
2	Read Setpoint	2	4	51	98	145
2	Alarm Code	1	6	53	100	147
2	Read Override	1	7	54	101	148
3	Full Scale	4	8	55	102	149
3	Gas Symbol	6	12	59	106	153
3	Read calibration Slot	1	18	65	112	159
4	MFC Serial Number	10	19	66	113	160
4	Product Identifier	10	29	76	123	170
4	Manufacturer Identifier	4	39	86	133	180
4	Software Version	4	43	90	137	184
TOTAL frame length			47	94	141	188

PROFIBUS outputs			Offset			
Class	Variable name	Length	MFC 1	MFC 2	MFC 3	MFC 4
1	Write Setpoint	2	0	5	10	15
2	Write Override	1	2	7	12	17
2	Reset Alarm	1	3	8	13	18
3	Write Calibration Slot	1	4	9	14	19
TOTAL frame length			5	10	15	20

9. Appendix 1: GSD file

GSD and DIB files are available on request.

DIB file contains the following icon :



```

;=====
; GSD File  QUALIFLOW MFC
; Modular Slave without parameters
;
; Version:  V1.2
;=====
#Profibus_DP
;General parameters
GSD_Revision      = 2
Vendor_Name       = "QUALIFLOW"
Model_Name        = "MFC"
Revision          = "V1.2"
Ident_Number      = 0x05B6
Protocol_Ident    = 0
Station_Type      = 0
FMS_supp          = 0
Hardware_Release  = "HW_1.0"
Software_Release  = "SW_1.0"
9.6_supp          = 1
19.2_supp         = 1
45.45_supp        = 1
93.75_supp        = 1
187.5_supp        = 1
500_supp          = 1
1.5M_supp         = 1
3M_supp           = 1
6M_supp           = 1
12M_supp          = 1
MaxTsdR_9.6       = 60
MaxTsdR_19.2      = 60
MaxTsdR_45.45     = 250
MaxTsdR_93.75     = 60
MaxTsdR_187.5     = 60
MaxTsdR_500       = 100
MaxTsdR_1.5M      = 150
MaxTsdR_3M        = 250
MaxTsdR_6M        = 450
MaxTsdR_12M       = 800
Redundancy        = 0
Repeater_Ctrl_Sig = 0
24V_Pins          = 0
Implementation_Type = "SPC3"
Bitmap_Device     = "Qual05b6"
Bitmap_Diag       = "Qual05b6"
Bitmap_SF         = "Qual05b6"
; Slave-Specification:
Freeze_Mode_supp = 1
Sync_Mode_supp   = 1
Set_Slave_Add_Supp = 0
Auto_Baud_supp   = 1
Min_Slave_Intervall = 1
Fail_Safe        = 0
Max_Diag_Data_Len = 6
Modul_Offset     = 0
Slave_Family     = 9@qualiflow
Modular_Station  = 1
Max_Module       = 1
Max_Input_len    = 188
Max_Output_len   = 20
Max_Data_len     = 208

```

```

; UserPrmData: Length and Preset:
User_Prm_Data_Len      = 0

Module="1 MFC datamin" 0xD1, 0xE0
1
EndModule

Module="2 MFC datamin" 0xD1, 0xE0, 0xD1, 0xE0
2
EndModule

Module="3 MFC datamin" 0xD1, 0xE0, 0xD1, 0xE0, 0xD1, 0xE0
3
EndModule

Module="4 MFC datamin" 0xD1, 0xE0, 0xD1, 0xE0, 0xD1, 0xE0, 0xD1, 0xE0
4
EndModule

Module="1 MFC datamin + dataext" 0xD3, 0xE1
5
EndModule

Module="2 MFC datamin + dataext" 0xD3, 0xE1, 0xD3, 0xE1
6
EndModule

Module="3 MFC datamin + dataext" 0xD3, 0xE1, 0xD3, 0xE1, 0xD3, 0xE1
7
EndModule

Module="4 MFC datamin + dataext" 0xD3, 0xE1, 0xD3, 0xE1, 0xD3, 0xE1, 0xD3, 0xE1
8
EndModule

Module="1 MFC datamin + dataext + config" 0xD8, 0xE1, 0xB0
9
EndModule

Module="2 MFC datamin + dataext + config" 0xD8, 0xE1, 0xB0, 0xD8, 0xE1, 0xB0
10
EndModule

Module="3 MFC datamin + dataext + config" 0xD8, 0xE1, 0xB0, 0xD8, 0xE1, 0xB0, 0xD8,
0xE1, 0xB0
11
EndModule

Module="4 MFC datamin + dataext + config" 0xD8, 0xE1, 0xB0, 0xD8, 0xE1, 0xB0, 0xD8,
0xE1, 0xB0, 0xD8, 0xE1, 0xB0
12
EndModule

Module="1 MFC datamin+dataext+config+id" 0xDF, 0xD6, 0xE1, 0xB0
13
EndModule

Module="2 MFC datamin+dataext+config+id" 0xDF, 0xD6, 0xE1, 0xB0, 0xDF, 0xD6, 0xE1, 0xB0
14
EndModule

Module="3 MFC datamin+dataext+config+id" 0xDF, 0xD6, 0xE1, 0xB0, 0xDF, 0xD6, 0xE1, 0xB0,
0xDF, 0xD6, 0xE1, 0xB0
15
EndModule

Module="4 MFC datamin+dataext+config+id" 0xDF, 0xD6, 0xE1, 0xB0, 0xDF, 0xD6, 0xE1, 0xB0,
0xDF, 0xD6, 0xE1, 0xB0, 0xDF, 0xD6, 0xE1, 0xB0
16
EndModule

```

10. Appendix 2 - Definitions

Address: The address is an absolute numerical reference to a parameter within a device.

Alert Objects: They are used to communicate notification messages when alarms or events are detected.

Application: A *software functional unit* consisting of an interconnected aggregation of *function blocks, events and objects*, which may be distributed and which may have *interfaces* with other *applications*.

Attribute: A property or characteristic of an *entity*; for instance, value and status are attributes of an output parameter.

Bus address: The bus address is the numerical reference of the device at the network.

Block (block instance): A logical processing unit of software comprising an individual, named copy of the block and associated parameters specified by a block type, which persists from one invocation of the block to the next.

Data structure: An *aggregate* whose elements need not be of the same *data type*, and each of which may be uniquely referenced by an *identifier*.

Data type: A set of values together with a set of permitted *operations*.

Entity: A particular thing, such as a person, place, *process*, object, concept, association or *event*.

Function: (1) A specific purpose of an entity. (2) One of a group of actions performed by an entity in accomplishing its purposes.

Function Block: A named *block* consisting of one or more input, output and contained parameters. Function blocks represent the basic automation functions performed by an application which is as independent as possible of the specifics of I/O devices and the network. Each function block processes input parameters according to a specified algorithm and an internal set of contained parameters. They produce output parameters that are available for use within the same function block application or by other function block applications.

Function Block application: application of a automation system performed by Physical Block, Function Block, Transducer Block and accompanied objects.

Instance: A piece of data related to an invocation of a function block.

MFC: Stands for Mass Flow Controller. This device can be analog or digital.

Mode: Determines the block operating mode and available modes for a block instance.

Object: An entity having state, behavior and identity.

Parameter: A *variable* that is an input, output or contained one of a function block.

Physical Block: Hardware specific characteristics of a field device, which are associated with a resource, are made visible through the physical block. Similar to transducer blocks, they insulate function blocks from the physical hardware by containing a set of implementation independent hardware parameters.

Record: A set of *data elements* treated as a unit.

Relative Index: The relative index is a logical offset of a parameter in a block.

Simple variable: A single variable which is characterized by a defined Data Type.

Transducer Block: Transducer blocks insulate function blocks from the specifics of I/O devices, such as sensors, actuators, and switches. Transducer blocks PROFIBUS-PA Profile for Process Control Devices, Version 3.0 General Requirements control access to I/O devices through a device independent interface defined for use by function blocks. Transducer blocks also perform functions, such as calibration and linearization, on I/O data to convert it to a device independent representation. Their interface to function blocks is defined as one or more implementation independent I/O channels.

Variable: A *software* entity that may assume any one of a set of values. The values of a variable are usually restricted to a certain data type.

View Objects: Provided to support efficient access to parameter data within a function block application. View objects allow groups of parameters to be accessed with a single communication request.

oOo