

=====  
KSUA MODBUS RTU INTERFACE - Commands and Responses  
=====

Document version: 2007-05-29  
KSUA Firmware version: 1.8

=====  
TABLE OF CONTENTS

READ DETECTOR STATUS (COMPACT) .....	2
READ DAMPER POSITIONS (COMPACT) .....	3
READ DETECTOR GROUP FIRE STATUS (COMPACT) .....	4
READ INPUTS AND FLAGS (COMPACT) .....	5
READ/WRITE REAL TIME CLOCK .....	6
READ DETECTOR STATUS (LONG) .....	8
READ DAMPER POSITIONS (LONG) .....	9
READ DETECTOR GROUP FIRE STATUS (LONG) .....	10
READ INPUTS AND FLAGS (LONG) .....	11
READ KSUC3 ALARM INPUTS (COMPACT) .....	12
READ KSUC3 ALARM INPUTS (LONG) .....	13
READ EXTERNAL DETECTOR STATUS (COMPACT).....	15
READ EXTERNAL DETECTOR STATUS (LONG).....	16
READ/WRITE VARIOUS CONTROL AND STATUS BITS ...	17
DIAGNOSTIC FUNCTION .....	18
EXCEPTION CODES .....	18

```
=====
READ DETECTOR STATUS COMPACT
=====
```

See also "READ DETECTOR STATUS LONG" below.

16 registers (16-bit)  
 4 detectors per register  
 \* = unused bits, read as zero

16-bit register read:  
 Function\_code: 0x04 ("Read Input Registers")  
 Start\_address: Register address, 0x0000..0x000F  
 Register\_count: 1..(16-Start\_Address\_LSByte)

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit number
0x0000	4	4	4		3	3	3		2	2	2		1	1	1		Det. number
	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	Det. flags
0x0001	8	8	8		7	7	7		6	6	6		5	5	5		
	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
0x000E	60	60	60		59	59	59		58	58	58		57	57	57		
	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	
0x000F	64	64	64		63	63	63		62	62	62		61	61	61		
	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	*	FAIL	SERV	FIRE	

Example MODBUS transaction for "READ DETECTOR STATUS COMPACT" (CRC not shown):

Master request:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Starting Address MSB = 0  
 Byte 3 = Starting Address LSB = 0..15  
 Byte 4 = Register Count MSB = 0  
 Byte 5 = Register Count LSB = 1..(16-Start\_Address\_LSByte)

Slave response:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Total Register Bytecount = 2 \* (Register Count) = 2..32  
 Byte 3 = Register MSB from starting address  
 Byte 4 = Register LSB from starting address  
 Byte 5 = Register MSB from starting address+1  
 Byte 6 = Register LSB from starting address+1  
 ...  
 ...  
 Byte ? = Register MSB from starting address+(Register Count-1)  
 Byte ? = Register LSB from starting address+(Register Count-1)

```
=====
READ DAMPER POSITIONS COMPACT
=====
```

See also "READ DAMPER POSITIONS LONG" below.

8 registers (16-bit)  
8 dampers per register

16-bit register read:

Function\_code: 0x04 ("Read Input Registers")  
Start\_address: Register address, 0x0100..0x0107  
Register\_count: 1..(8-Start\_Address\_LSByte)

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit number
0x0100	8	8	7	7	6	6	5	5	4	4	3	3	2	2	1	1	< Damper num. < Pos. flags
	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	
0x0101	16	16	15	15	14	14	13	13	12	12	11	11	10	10	9	9	
	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
0x0106	56	56	55	55	54	54	53	53	52	52	51	51	50	50	49	49	
	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	
0x0107	64	64	63	63	62	62	61	61	60	60	59	59	58	58	57	57	
	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	

Example MODBUS transaction for "READ DAMPER POSITIONS COMPACT" (CRC not shown):

Master request:

Byte 0 = Slave Address = 1..247  
Byte 1 = Function Code = 4  
Byte 2 = Starting Address MSB = 1  
Byte 3 = Starting Address LSB = 0..7  
Byte 4 = Register Count MSB = 0  
Byte 5 = Register Count LSB = 1..(8-Start\_Address\_LSByte)

Slave response:

Byte 0 = Slave Address = 1..247  
Byte 1 = Function Code = 4  
Byte 2 = Total Register Bytecount = 2 \* (Register Count) = 2..16  
Byte 3 = Register MSB from starting address  
Byte 4 = Register LSB from starting address  
Byte 5 = Register MSB from starting address+1  
Byte 6 = Register LSB from starting address+1  
...  
...  
Byte ? = Register MSB from starting address+(Register Count-1)  
Byte ? = Register LSB from starting address+(Register Count-1)

```
=====
READ DETECTOR GROUP FIRE STATUS COMPACT
=====
```

See also "READ DETECTOR GROUP FIRE STATUS LONG" below.

4 registers (16-bit)  
16 detector groups per register

16-bit register read:  
Function\_code: 0x04 ("Read Input Registers")  
Start\_address: Register address, 0x0200..0x0203  
Register\_count: 1..(4-Start\_Address\_LSByte)

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	< Bit number
0x0200	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	< Group num.
	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	< Fire flags
0x0201	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	
	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	
0x0202	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	
	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	
0x0203	64	63	62	61	44	43	42	41	40	39	38	37	36	35	34	33	
	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	FIRE	

Example MODBUS transaction for "READ DETECTOR GROUP FIRE STATUS COMPACT" (CRC not shown):

Master request:  
Byte 0 = Slave Address = 1..247  
Byte 1 = Function Code = 4  
Byte 2 = Starting Address MSB = 2  
Byte 3 = Starting Address LSB = 0..3  
Byte 4 = Register Count MSB = 0  
Byte 5 = Register Count LSB = 1..(4-Start\_Address\_LSByte)

Slave response:  
Byte 0 = Slave Address = 1..247  
Byte 1 = Function Code = 4  
Byte 2 = Total Register Bytecount = 2 \* (Register Count) = 2..8  
Byte 3 = Register MSB from starting address  
Byte 4 = Register LSB from starting address  
Byte 5 = Register MSB from starting address+1  
Byte 6 = Register LSB from starting address+1  
...  
...  
Byte ? = Register MSB from starting address+(Register Count-1)  
Byte ? = Register LSB from starting address+(Register Count-1)

```
=====
READ INPUTS AND FLAGS COMPACT
=====
```

See also "READ INPUTS AND FLAGS LONG" below.

1 register (16-bit)  
 \* = unused bits, read as zero

16-bit register read:  
 Function\_code: 0x04 ("Read Input Registers")  
 Start\_address: Register address, 0x0300  
 Register\_count: 1

Flag names:

- "FIRE ALRM" = Fire Alarm Relay, 1 = Activated (Fire)
- "SUM ALRM" = Sum Alarm Relay, 1 = Activated (Fault or Fire)
- "VENT FAN" = Power Relay 1 (always VENT Fan), 1 = Relay ON
- "POWR REL2" = Power Relay 2 (VENT Fan, EVAC Fan, Heater or none), 1 = Relay ON
- "EXT ALRM" = External Alarm Input on KSUA, 1 = Activated (Fire alarm!)
- "FORC OPEN" = Forced Opening Input on KSUA, 1 = Activated (Open!)
- "EXT NITE" = External Night Input on KSUA, 1 = Activated (Night!)
- "SLAVE DAY" = KSUA forced to daytime mode by request from KSUB slave (if flag = 1)
- "NITE FLAG" = Current KSUA mode, 1 = Night, 0 = Day
- "DMPR TEST" = Damper test, and possibly also EVAC fan test, is in progress (if flag = 1)

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	< Bit number
0x0300	*	*	*	*	*	*	*	DMPR	NITE	SLAV	EXT	FORC	EXT	POWR	VENT	SUM	FIRE
								TEST	FLAG	DAY	NITE	OPEN	ALRM	REL2	FAN	ALRM	ALRM

Example MODBUS transaction for "READ INPUTS AND FLAGS COMPACT" (CRC not shown):

Master request:

- Byte 0 = Slave Address = 1..247
- Byte 1 = Function Code = 4
- Byte 2 = Starting Address MSB = 3
- Byte 3 = Starting Address LSB = 0
- Byte 4 = Register Count MSB = 0
- Byte 5 = Register Count LSB = 1

Slave response:

- Byte 0 = Slave Address = 1..247
- Byte 1 = Function Code = 4
- Byte 2 = Total Register Bytecount = 2
- Byte 3 = Register at 0x0300, MSB
- Byte 4 = Register at 0x0300, LSB

```
=====
READ/WRITE REAL TIME CLOCK
=====
```

4 registers (16-bit)  
 \* = unused bits, read/write as zero

All values are in binary format (not BCD!):  
 SEC5..SEC0 = Seconds, 0..59  
 MIN5..MIN0 = Minutes, 0..59  
 HOUR4..HOUR0 = Hours, 0..23  
 WDAY2..WDAY0 = Weekday, 0..6 where 0 is Sunday  
 DATE4..DATE0 = Date, 1..31  
 MONT3..MONT0 = Month, 1..12  
 YEAR6..YEAR0 = Year, 0..99

16-bit register write:  
 Function\_code: 0x10 ("Write Multiple Registers")  
 Start\_address: Register address, 0x0400  
 Register\_count: 4 (must be 4!)

16-bit register read:  
 Function\_code: 0x04 ("Read Input Registers")  
 Start\_address: Register address, 0x0400  
 Register\_count: 4 (must be 4!)

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	< Bit number
0x0400	*	*	MIN	MIN	MIN	MIN	MIN	MIN	*	*	SEC	SEC	SEC	SEC	SEC	SEC	
			5	4	3	2	1	0			5	4	3	2	1	0	
0x0401	*	*	*	*	*	WDAY	WDAY	WDAY	*	*	*	HOUR	HOUR	HOUR	HOUR	HOUR	
						2	1	0				4	3	2	1	0	
0x0402	*	*	*	*	MONT	MONT	MONT	MONT	*	*	*	DATE	DATE	DATE	DATE	DATE	
					3	2	1	0				4	3	2	1	0	
0x0403	*	*	*	*	*	*	*	*	*	YEAR	YEAR	YEAR	YEAR	YEAR	YEAR	YEAR	
										6	5	4	3	2	1	0	

Example MODBUS transaction for "READ REAL TIME CLOCK" (CRC not shown):

Master request:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Starting Address MSB = 4  
 Byte 3 = Starting Address LSB = 0  
 Byte 4 = Register Count MSB = 0  
 Byte 5 = Register Count LSB = 4

Slave response:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Total Register Bytecount = 8  
 Byte 3 = Register at 0x0400, MSB  
 Byte 4 = Register at 0x0400, LSB  
 Byte 5 = Register at 0x0401, MSB  
 Byte 6 = Register at 0x0401, LSB  
 Byte 7 = Register at 0x0402, MSB  
 Byte 8 = Register at 0x0402, LSB  
 Byte 9 = Register at 0x0403, MSB  
 Byte10 = Register at 0x0403, LSB

(continued on next page)

```
=====
READ/WRITE REAL TIME CLOCK (continued)
=====
```

Example MODBUS transaction for "WRITE REAL TIME CLOCK" (CRC not shown):

Master request:

```
Byte 0 = Slave Address = 1..247
Byte 1 = Function Code = 16 (0x10)
Byte 2 = Starting Address MSB = 4
Byte 3 = Starting Address LSB = 0
Byte 4 = Register Count MSB = 0
Byte 5 = Register Count LSB = 4
Byte 6 = Total Register Bytecount = 8
Byte 7 = MINUTES (0..59 in binary format)
Byte 8 = SECONDS (0..59 in binary format)
Byte 9 = WEEKDAY (0..6 in binary format)
Byte10 = HOUR (0..23 in binary format)
Byte11 = MONTH (1..12 in binary format)
Byte12 = DATE (1..31 in binary format)
Byte13 = 0 (not used but keep this byte cleared for future compatibility!)
Byte14 = YEAR (0..99 in binary format)
```

Slave response:

```
Byte 0 = Slave Address = 1..247
Byte 1 = Function Code = 16 (0x10)
Byte 2 = Starting Address MSB = 4
Byte 3 = Starting Address LSB = 0
Byte 4 = Register Count MSB = 0
Byte 5 = Register Count LSB = 4
```

```
=====
READ DETECTOR STATUS LONG
=====
```

See also "READ DETECTOR STATUS LONG COMPACT" above.

64 registers (16-bit)  
 1 detector per register  
 \* = unused bits, read as zero  
 Detector flag polarity: Logic '1' = TRUE

16-bit register read:  
 Function\_code: 0x04 ("Read Input Registers")  
 Start\_address: Register address, 0x0500..0x053F  
 Register\_count: 1..(64-Start\_Address\_LSByte)

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit number
0x0500	*	*	*	*	*	*	*	*	*	*	*	*	*	1	1	1	Det. number
	*	*	*	*	*	*	*	*	*	*	*	*	*	FAIL	SERV	FIRE	Det. flags
0x0501	*	*	*	*	*	*	*	*	*	*	*	*	*	2	2	2	
	*	*	*	*	*	*	*	*	*	*	*	*	*	FAIL	SERV	FIRE	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
0x053E	*	*	*	*	*	*	*	*	*	*	*	*	*	63	63	63	
	*	*	*	*	*	*	*	*	*	*	*	*	*	FAIL	SERV	FIRE	
0x053F	*	*	*	*	*	*	*	*	*	*	*	*	*	64	64	64	
	*	*	*	*	*	*	*	*	*	*	*	*	*	FAIL	SERV	FIRE	

Example MODBUS transaction for "READ DETECTOR STATUS LONG" (CRC not shown):

Master request:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Starting Address MSB = 5  
 Byte 3 = Starting Address LSB = 0..63  
 Byte 4 = Register Count MSB = 0  
 Byte 5 = Register Count LSB = 1..(64-Start\_Address\_LSByte)

Slave response:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Total Register Bytecount = 2 \* (Register Count) = 2..128  
 Byte 3 = Register MSB from starting address  
 Byte 4 = Register LSB from starting address  
 Byte 5 = Register MSB from starting address+1  
 Byte 6 = Register LSB from starting address+1  
 ...  
 ...  
 Byte ? = Register MSB from starting address+(Register Count-1)  
 Byte ? = Register LSB from starting address+(Register Count-1)

```
=====
READ DAMPER POSITIONS LONG
=====
```

See also "READ DAMPER POSITIONS COMPACT" above.

64 registers (16-bit)  
 1 damper per register  
 \* = unused bits, read as zero  
 Damper flag polarity: Logic '1' = TRUE

16-bit register read:  
 Function\_code: 0x04 ("Read Input Registers")  
 Start\_address: Register address, 0x0600..0x063F  
 Register\_count: 1..(64-Start\_Address\_LSByte)

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit number
0x0600	*	*	*	*	*	*	*	*	*	*	*	*	*	*	ON	OFF	< Damper num. < Pos. flags
0x0601	*	*	*	*	*	*	*	*	*	*	*	*	*	*	ON	OFF	2 2
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
0x063E	*	*	*	*	*	*	*	*	*	*	*	*	*	*	ON	OFF	63 63
0x063F	*	*	*	*	*	*	*	*	*	*	*	*	*	*	ON	OFF	64 64

Example MODBUS transaction for "READ DAMPER POSITIONS LONG" (CRC not shown):

Master request:

Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Starting Address MSB = 6  
 Byte 3 = Starting Address LSB = 0..63  
 Byte 4 = Register Count MSB = 0  
 Byte 5 = Register Count LSB = 1..(64-Start\_Address\_LSByte)

Slave response:

Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Total Register Bytecount = 2 \* (Register Count) = 2..128  
 Byte 3 = Register MSB from starting address  
 Byte 4 = Register LSB from starting address  
 Byte 5 = Register MSB from starting address+1  
 Byte 6 = Register LSB from starting address+1  
 ...  
 ...  
 Byte ? = Register MSB from starting address+(Register Count-1)  
 Byte ? = Register LSB from starting address+(Register Count-1)

```
=====
READ DETECTOR GROUP FIRE STATUS LONG
=====
```

See also "READ DETECTOR GROUP FIRE STATUS COMPACT" above.

64 registers (16-bit)  
 1 detector group per register  
 \* = unused bits, read as zero

16-bit register read:  
 Function\_code: 0x04 ("Read Input Registers")  
 Start\_address: Register address, 0x0700..0x073F  
 Register\_count: 1..(64-Start\_Address\_LSByte)

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit number
0x0700	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	1	Group num. Fire flag
0x0701	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	2	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
0x073E	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	63	
0x073F	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	64	

Example MODBUS transaction for "READ DETECTOR GROUP FIRE STATUS LONG" (CRC not shown):

Master request:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Starting Address MSB = 7  
 Byte 3 = Starting Address LSB = 0..63  
 Byte 4 = Register Count MSB = 0  
 Byte 5 = Register Count LSB = 1..(64-Start\_Address\_LSByte)

Slave response:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Total Register Bytecount = 2 \* (Register Count) = 2..128  
 Byte 3 = Register MSB from starting address  
 Byte 4 = Register LSB from starting address  
 Byte 5 = Register MSB from starting address+1  
 Byte 6 = Register LSB from starting address+1  
 ...  
 ...  
 Byte ? = Register MSB from starting address+(Register Count-1)  
 Byte ? = Register LSB from starting address+(Register Count-1)

```
=====
READ INPUTS AND FLAGS LONG
=====
```

See also "READ INPUTS AND FLAGS COMPACT" above.

10 registers (16-bit)  
 \* = unused bits, read as zero

16-bit register read:  
 Function\_code: 0x04 ("Read Input Registers")  
 Start\_address: Register address, 0x0800..0x809  
 Register\_count: 1..(10-Start\_Address\_LSByte)

Flag names:  
 "FIRE ALRM" = Fire Alarm Relay, 1 = Activated (Fire)  
 "SUM ALRM" = Sum Alarm Relay, 1 = Activated (Fault or Fire)  
 "VENT FAN" = Power Relay 1 (always VENT Fan), 1 = Relay ON  
 "POWR REL2" = Power Relay 2 (VENT Fan, EVAC Fan, Heater or none), 1 = Relay ON  
 "EXT ALRM" = External Alarm Input on KSUA, 1 = Activated (Fire alarm!)  
 "FORC OPEN" = Forced Opening Input on KSUA, 1 = Activated (Open!)  
 "EXT NITE" = External Night Input on KSUA, 1 = Activated (Night!)  
 "SLAVE DAY" = KSUA forced to daytime mode by request from KSUB slave (if flag = 1)  
 "NITE FLAG" = Current KSUA mode, 1 = Night, 0 = Day  
 "DMPR TEST" = Damper test, and possibly also EVAC fan test, is in progress (if flag = 1)

Reg. Addr.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	< Bit number
0x0800	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FIRE ALRM
0x0801	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	SUM ALRM
0x0802	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	VENT FAN
0x0803	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	POWR REL2
0x0804	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	EXT ALRM
0x0805	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FORC OPEN
0x0806	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	EXT NITE
0x0807	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	SLAV DAY
0x0808	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	NITE FLAG
0x0809	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	DMPR TEST

Example MODBUS transaction for "READ INPUTS AND FLAGS LONG" (CRC not shown):

Master request:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Starting Address MSB = 8  
 Byte 3 = Starting Address LSB = 0..9  
 Byte 4 = Register Count MSB = 0  
 Byte 5 = Register Count LSB = 1..(10-Start\_Address\_LSByte)

Slave response:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Total Register Bytecount = 2 \* (Register Count) = 2..20  
 Byte 3 = Register MSB from starting address  
 Byte 4 = Register LSB from starting address  
 Byte 5 = Register MSB from starting address+1  
 Byte 6 = Register LSB from starting address+1  
 ...  
 ...  
 Byte ? = Register MSB from starting address+(Register Count-1)  
 Byte ? = Register LSB from starting address+(Register Count-1)

```
=====
READ KSUC3 ALARM INPUTS COMPACT
=====
```

See also "READ KSUC3 ALARM INPUTS LONG" below.

1 register (16-bit)

16-bit register read:

```
Function_code: 0x04 ("Read Input Registers")
Start_address: Register address, 0x0900
Register_count: 1
```

KSUC3 Alarm Flags:

```
1 = Alarm
0 = No Alarm
```

```
|-----|
| Reg. |-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Addr. | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | < Bit number
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|0x0900|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|FLAG|
|      |#16|#15|#14|#13|#12|#11|#10|#9|#8|#7|#6|#5|#4|#3|#2|#1|
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
```

Example MODBUS transaction for "READ KSUC3 ALARM INPUTS COMPACT" (CRC not shown):

Master request:

```
Byte 0 = Slave Address = 1..247
Byte 1 = Function Code = 4
Byte 2 = Starting Address MSB = 9
Byte 3 = Starting Address LSB = 0
Byte 4 = Register Count MSB = 0
Byte 5 = Register Count LSB = 1
```

Slave response:

```
Byte 0 = Slave Address = 1..247
Byte 1 = Function Code = 4
Byte 2 = Total Register Bytecount = 2
Byte 3 = Register at 0x0900, MSB
Byte 4 = Register at 0x0900, LSB
```

```
=====
READ KSUC3 ALARM INPUTS LONG
=====
```

See also "READ KSUC3 ALARM INPUTS COMPACT" above.

16 registers (16-bit)  
 \* = unused bits, read as zero

16-bit register read:  
 Function\_code: 0x04 ("Read Input Registers")  
 Start\_address: Register address, 0x0A00..0xA0F  
 Register\_count: 1..(16-Start\_Address\_LSByte)

KSUC3 Alarm Flags:  
 1 = Alarm  
 0 = No Alarm

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	< Bit number
0x0A00	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #1
0x0A01	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #2
0x0A02	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #3
0x0A03	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #4
0x0A04	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #5
0x0A05	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #6
0x0A06	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #7
0x0A07	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #8
0x0A08	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #9
0x0A09	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #10
0x0A0A	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #11
0x0A0B	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #12
0x0A0C	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #13
0x0A0D	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #14
0x0A0E	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #15
0x0A0F	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	FLAG #16

(continued on next page)

```
=====
READ KSUC3 ALARM INPUTS LONG (continued)
=====
```

Example MODBUS transaction for "READ KSUC3 ALARM INPUTS LONG" (CRC not shown):

Master request:

- Byte 0 = Slave Address = 1..247
- Byte 1 = Function Code = 4
- Byte 2 = Starting Address MSB = 10 (0x0A)
- Byte 3 = Starting Address LSB = 0..15
- Byte 4 = Register Count MSB = 0
- Byte 5 = Register Count LSB = 1..(16-Start\_Address\_LSByte)

Slave response:

- Byte 0 = Slave Address = 1..247
- Byte 1 = Function Code = 4
- Byte 2 = Total Register Bytecount = 2 \* (Register Count) = 2..32
- Byte 3 = Register MSB from starting address
- Byte 4 = Register LSB from starting address
- Byte 5 = Register MSB from starting address+1
- Byte 6 = Register LSB from starting address+1
- ...
- ...
- Byte ? = Register MSB from starting address+(Register Count-1)
- Byte ? = Register LSB from starting address+(Register Count-1)

```
=====
READ EXTERNAL DETECTOR STATUS COMPACT
=====
```

See also "READ EXTERNAL DETECTOR STATUS LONG" below.

2 registers (16-bit)

16-bit register read:

```
Function_code: 0x04 ("Read Input Registers")
Start_address: Register address, 0x0B00..0x0B01
Register_count: 1..(2-Start_Address_LSByte)
```

External detector flags:

```
1 = Alarm
0 = No Alarm
```

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit number
0x0B00	80e	79e	78e	77e	76e	75e	74e	73e	72e	71e	70e	69e	68e	67e	66e	65e	From KSUC1
0x0B01	96e	95e	94e	93e	92e	91e	90e	89e	88e	87e	86e	85e	84e	83e	82e	81e	From KSUC2

Example MODBUS transaction for "READ EXTERNAL DETECTOR STATUS COMPACT" (CRC not shown):

Master request:

```
Byte 0 = Slave Address = 1..247
Byte 1 = Function Code = 4
Byte 2 = Starting Address MSB = 11 (0x0B)
Byte 3 = Starting Address LSB = 0
Byte 4 = Register Count MSB = 0..1
Byte 5 = Register Count LSB = 1..(2-Start_Address_LSByte)
```

Slave response:

```
Byte 0 = Slave Address = 1..247
Byte 1 = Function Code = 4
Byte 2 = Total Register Bytecount = 2 * (Register Count) = 2..4
Byte 3 = Register MSB from starting address
Byte 4 = Register LSB from starting address
If Register Count = 2:
  Byte 5 = Register MSB from starting address+1
  Byte 6 = Register LSB from starting address+1
```

```
=====
READ EXTERNAL DETECTOR STATUS LONG
=====
```

See also "READ EXTERNAL DETECTOR STATUS COMPACT" above.

32 registers (16-bit)  
 \* = unused bits, read as zero

16-bit register read:  
 Function\_code: 0x04 ("Read Input Registers")  
 Start\_address: Register address, 0x0C00..0xC1F  
 Register\_count: 1..(32-Start\_Address\_LSByte)

External detector (KSUC1, KSUC2) alarm Flags:  
 1 = Alarm  
 0 = No Alarm

Reg.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	< Bit number	
0x0C00	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	65e FIRE	< Ext. det. number
0x0C01	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	66e FIRE	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.....	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
0x0C1E	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	95e FIRE	
0x0C1F	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	96e FIRE	

Example MODBUS transaction for "READ EXTERNAL DETECTOR STATUS LONG" (CRC not shown):

Master request:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Starting Address MSB = 12 (0x0C)  
 Byte 3 = Starting Address LSB = 0..31  
 Byte 4 = Register Count MSB = 0  
 Byte 5 = Register Count LSB = 1..(32-Start\_Address\_LSByte)

Slave response:  
 Byte 0 = Slave Address = 1..247  
 Byte 1 = Function Code = 4  
 Byte 2 = Total Register Bytecount = 2 \* (Register Count) = 2..64  
 Byte 3 = Register MSB from starting address  
 Byte 4 = Register LSB from starting address  
 Byte 5 = Register MSB from starting address+1  
 Byte 6 = Register LSB from starting address+1  
 ...  
 ...  
 Byte ? = Register MSB from starting address+(Register Count-1)  
 Byte ? = Register LSB from starting address+(Register Count-1)

```
=====
READ/WRITE VARIOUS CONTROL AND STATUS BITS
=====
```

```
Single bit write (MODBUS Nightmode Control only):
  Function_code: 0x05 ("Write Single Coil")
  Output_address: 0x0000
  Output_value: 0xFF00 for NIGHT, 0x0000 for DAY
```

```
Single bit read (MODBUS Nightmode Control only):
  Function_code: 0x01 ("Read Coils")
  Start_address: 0x0000
  Bit_count: 1
```

```
Single bit write (Start Damper Test only):
  Function_code: 0x05 ("Write Single Coil")
  Output_address: 0x0001
  Output_value: 0xFF00
```

```
Single bit write (Alarm Reset only):
  Function_code: 0x05 ("Write Single Coil")
  Output_address: 0x0002
  Output_value: 0xFF00
```

Bit address	Bit function	Read	Write
0x0000	MODBUS Nightmode control	YES	YES
0x0001	Start Damper Test	NO	YES
0x0002	Alarm Reset	NO	YES

Example MODBUS transaction for "MODBUS nightmode control write" (CRC not shown):

```
Master request:
  Byte 0 = Slave Address = 1..247
  Byte 1 = Function Code = 5
  Byte 2 = Output Address MSB = 0
  Byte 3 = Output Address LSB = 0
  Byte 4 = Output Value MSB = 0 for "DAY", 255 for "NIGHT"
  Byte 5 = Output Value LSB = 0
```

```
Slave response:
  Exactly the same as the master request.
```

Example MODBUS transaction for "Start damper test write" (CRC not shown):

```
Master request:
  Byte 0 = Slave Address = 1..247
  Byte 1 = Function Code = 5
  Byte 2 = Output Address MSB = 0
  Byte 3 = Output Address LSB = 1
  Byte 4 = Output Value MSB = 255 (0 is also a valid value but does *NOT* start the test!)
  Byte 5 = Output Value LSB = 0
```

```
Slave response:
  Exactly the same as the master request.
```

Example MODBUS transaction for "Alarm reset write" (CRC not shown):

```
Master request:
  Byte 0 = Slave Address = 1..247
  Byte 1 = Function Code = 5
  Byte 2 = Output Address MSB = 0
  Byte 3 = Output Address LSB = 2
  Byte 4 = Output Value MSB = 255 (0 is also a valid value but does *NOT* reset the alarm!)
  Byte 5 = Output Value LSB = 0
```

```
Slave response:
  Exactly the same as the master request.
```

(continued on next page)

=====  
READ/WRITE VARIOUS CONTROL AND STATUS BITS (continued)  
=====

Example MODBUS transaction for "MODBUS nightmode control read" (CRC not shown):

Master request:

Byte 0 = Slave Address = 1..247  
Byte 1 = Function Code = 1  
Byte 2 = Starting Address MSB = 0  
Byte 3 = Starting Address LSB = 0  
Byte 4 = Number of bits MSB = 0  
Byte 5 = Number of bits LSB = 1

Slave response:

Byte 0 = Slave Address = 1..247  
Byte 1 = Function Code = 1  
Byte 2 = Bytecount = 1  
Byte 3 = MODBUS Nightmode Flag in bit 0 (remaining bits are zero)

=====  
DIAGNOSTIC FUNCTION  
=====

Diagnostic sub-function 0 (Return Query Data = ECHO message)  
is available for "pinging" the KSUA unit.

Example MODBUS transaction for "Return Query Data" (CRC not shown):

Master request:

Byte 0 = Slave Address = 1..247  
Byte 1 = Function Code = 8  
Byte 2 = Sub-function code MSB = 0  
Byte 3 = Sub-function code LSB = 0  
Byte 4..253 = any number (0..250) of bytes

Slave response:

Exactly the same as the master request.

=====  
EXCEPTION CODES  
=====

The following exception responses are implemented:

Exception code 1 - Illegal Function Code  
Exception code 2 - Illegal Data Address  
Exception code 3 - Illegal Data Value

<eof>