

# MatDeck Easy SCADA with ICP DAS Devices

MatDeck Easy SCADA offers users the ability to build industrial applications with ICP acquisition devices. The MatDeck Easy SCADA is easy to use as well as being able to integrate with ICP DAS configuration forms. Here, we illustrate how Easy SCADA can be used through a simple scenario using two ICP DAS devices, M-7026, and USB-2019, and virtual instruments provided by MatDeck.

## ICP Configuration using GUI Forms

The first steps in building SCADA applications is device selection and channel configuration. Here, we use M-7026, and we initiate `icpcom_multifunction7000_form()` in order to configure AO0, AO1, AI0 and AI1. The form generates a .txt files which is imported by SCADA and tags are generated automatically.

```
m7026 := icpcom_multifunction7000_form(o, "Form")
```

ICP DAS Multifunction Series 7000 Configuration Form: M-7002, M-7003, M-7026, and M-7024U

Module Selection

Device List - Address and ID

01 7026

Module Configuration

Address: 1 Fast Mode: Normal

Analog Format: Engineering

Response Delay: 0 ms Filter: 60Hz Rejection

INIT Configuration

Protocol: DCON Parity: N,8,1

Baud Rate: 9600 Checksum: Disable

AI(0:3) AI(4:7) AO(0:3) DO(0:3) DI(0:4) Host WDT

Ch 0

Enable

Type: -5~5 V

Alarm Mode: Disable

High Alarm Limit: 5.00

Low Alarm Limit: -5.00

Use as SCADA Tag

Ch 1

Enable

Type: -5~5 V

Alarm Mode: Disable

High Alarm Limit: 5.00

Low Alarm Limit: -5.00

Use as SCADA Tag

Ch 2

Enable

Type: -10~10 V

Alarm Mode: Disable

High Alarm Limit: 0.00

Low Alarm Limit: 0.00

Use as SCADA Tag

Ch 3

Enable

Type: -10~10 V

Alarm Mode: Disable

High Alarm Limit: 0.00

Low Alarm Limit: 0.00

Use as SCADA Tag

Back to Search Configure

In a similar manner, we use `icpusb_2019_2026_form()` to configure AI0 and AI1 of USB-2019. The configuration is exported to a .txt file, and used by SCADA to generate tags.

```
usb2019 := icpushb_2019_2026_form(o, "usb")
```

ICP DAS 2000 Series Configuration Form: 2019; 2026

Device Type and Board ID

USB2019 Board\_ID 1

USB2019

Device Settings

Filter Rejection: 60 Hz

Set Software WDT: 100.00 ms

Wire Detection: Enable

CJC for all Channels

CJC Offset: 0.00 °C

Enable/Disable: [Dropdown]

AI(0:3) AI(4:7)

AI0	AI1	AI2	AI3
Type: -5~+5 V	Type: -5~+5 V	Type: -15~+15 mV	Type: -15~+15 mV
Channel Enable: Enable	Channel Enable: Enable	Channel Enable: Enable	Channel Enable: Enable
CJC Offset: 0.00 °C	CJC Offset: 0.00 °C	CJC Offset: 0.00 °C	CJC Offset: 0.00 °C
<input checked="" type="checkbox"/> Use as SCADA Tag	<input checked="" type="checkbox"/> Use as SCADA Tag	<input type="checkbox"/> Use as SCADA Tag	<input type="checkbox"/> Use as SCADA Tag

Configure

The configuration functions are used to generate configuration files:

```
1 //icpcom_multifunction7000_form_configure(m7026)
2 //icpushb_2019_2026_form_configure(usb2019)
```

## SCADA Configuration

The first step is to define the database which will store the SCADA tags and relevant information. Next, the SCADA form is initiated by using the `scada_form()` function as seen in line two. The SCADA form is a intuitive GUI to build SCADA applications as illustrated below.

```
3 base := doc_dir() + "/" + "ScadaExampleMP.db"
4 scada := scada_form(0, "scada1", base)
```

Choose Devices
New Project
Open Project

Active Project  
 DB: ScadaExampleMP.db

TAG Table | TAG Data

Channel | ICPDAS | Instrument

Tag Name	Direction	Description	Min. Value	Max. Value	Sampling Rate	Unit
<input type="text"/>	Read ▼	<input type="text"/>	0.00 ▲▼	10.00 ▲▼	50 ms ▼	<input type="text"/>

Create Tag

Sel.	Enable	Tag Name	Direction	Min. Value	Max. Value	Sampling Rate	Description	Unit
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Slider1	Read ▼	0.00 ▲▼	10.00 ▲▼	200 ms ▼	<input type="text"/>	<input type="text"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Slider2	Read ▼	0.00 ▲▼	10.00 ▲▼	200 ms ▼	<input type="text"/>	<input type="text"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Indicator2	Write ▼				<input type="text"/>	<input type="text"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Indicator3	Write ▼				<input type="text"/>	<input type="text"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Switch1	Read ▼	0.00 ▲▼	10.00 ▲▼	200 ms ▼	<input type="text"/>	<input type="text"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Graph0	Write ▼				<input type="text"/>	<input type="text"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Graph1	Write ▼				<input type="text"/>	<input type="text"/>

Delete Selected Instruments Tags
Save Instrument Changes

SCADA Code

SCADA Variable:

Get Code

Stop

Expand Tags Table

## SCADA Scenario

In the SCADA Scenario under consideration, ICP M-7026 is used as a signal generator. The channel AO0 is used to generate a sinusoidal signal, and channel AO1 is used to generate a sawtooth signal. The frequency of the sinusoid is controlled by using Slider1, while the frequency of the sawtooth at AO1 is controlled by Slider 2. The analog input channels, AI0 and AI1, are connected to AO0 and AO1 respectively. AI0 and AI1 are also connected to Gauge1 and Gauge2, which show the current value of the read signal. The values of AO0 and AO1 are displayed in Graph1 and Graph2.

USB-2019 AI0 is used to follow M-7026 AO0, and USB-2019 AI1 is used to read M-7026 AO1. USB-2019 AI0 is also connected to Digit meter1, and USB-2019 AI1 is connected to Digit meter2, which shows the current values.

Additionally, there are two indicators, Indicator2 and Indicator3, which are related to events:

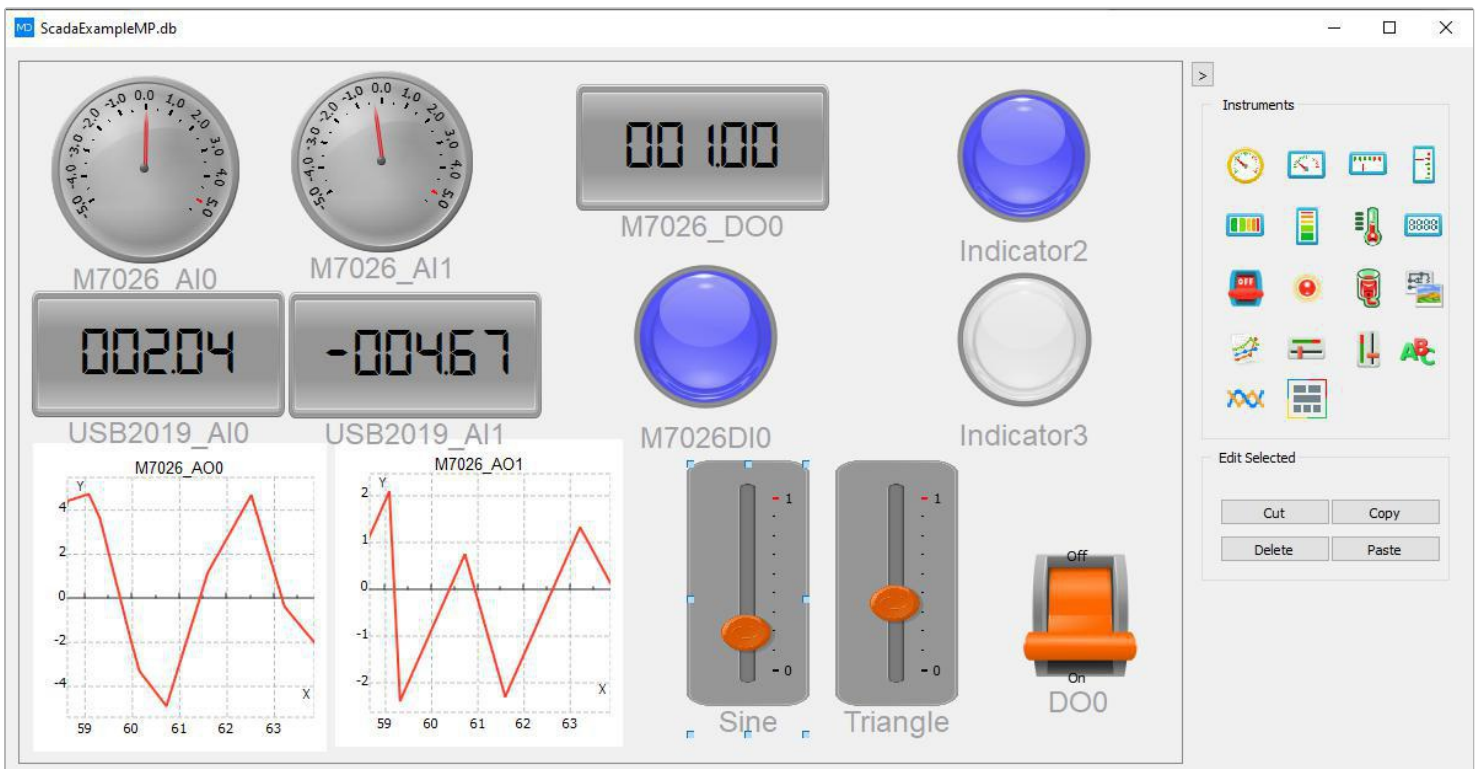
- If  $2V < M-7026\ A0 < 3V$  Indicator2 ON
- If  $0V < M-7026\ A1 < 3V$  Indicator3 ON

Finally, the overall SCADA process is controlled by using a switch connected with DO0 at M-7026. The signal from DO0 is transferred to DI0, and displayed at the indicator M7026DI0. The Scenario is displayed in table. The scenario is also implemented in a SCADA Panel and MatDeck script code.

Device, Channel	Source Device, Channel	Virtual Instrument
M-7026 AI0	M-7026 AO0	Gauge1
M-7026 AI1	M-7026 AO1	Gauge2
USB-2019 AI0	M-7026 AO0	Digit meter1
USB-2019 AI1	M-7026 AO1	Digit meter 2
M-7026 AO0	Slider1	Graph1
M-7026 AO1`	Slider2	Graph2
M-7026 DO0	Switch	
M-7026 DI0	M-7026 DO0	Indicator - M7026DI0

## SCADA Panel

The SCADA application defined above is implemented in the SCADA Panel. A image version of the SCADA Panel is shown below.



## Script Code

The code which is used to implement the SCADA application scenario is shown below.

```
5 T0 := timenow()
6 Frequency1 := 0
7 Frequency2 := 0
8 t := timer_create(200)
9 scada_start(scada)
10 vec1 := vector_create(10, false, 0)
11 vec2 := vector_create(10, false, 0)
12 tim := vector_create(10, false, 0)
13 count := 0
14
15 scada_script()
16 {
17     if(!scada_is_working(scada))
18     {
19         timer_delete(t)
20         return(void)
21     }
22
23     Switch1 := scada_tag_value(scada, "Switch1")
24     scada_tag_write_value(scada, "M7026_D00", Switch1)
25     M7026_DI0 := scada_tag_value(scada, "M7026_DI0")
26     if(Switch1 == 1)
27     {
28         Slider1 := scada_tag_value(scada, "Slider1")
29         Slider2 := scada_tag_value(scada, "Slider2")
30         Frequency1 = Slider1
31         Frequency2 = Slider2
32         currtime := timenow() - T0
33         sigs := sin(2 * cpi() * Frequency1 * currtime)
34         sawt := currtime * Frequency2 - floor(currtime * Frequency2 + 0.5)
35         tim[count] = currtime
36         vec1[count] = 5 * sigs
37         vec2[count] = 5 * sawt
38         //count += 1
39
40         if(count < 9)
41         {
42             count += 1
43         }
44         else
45         {
46             gr1 := join_mat_cols(tim, vec1)
47             gr2 := join_mat_cols(tim, vec2)
48             scada_tag_write_value(scada, "Graph0", gr1)
49             scada_tag_write_value(scada, "Graph1", gr2)
50             count = 0
51         }
52         scada_tag_write_value(scada, "M7026_A00", 5 * sigs)
53         scada_tag_write_value(scada, "M7026_A01", 5 * sawt)
54
55         USB2019_AI0 := scada_tag_value(scada, "USB2019_AI0")
56         USB2019_AI1 := scada_tag_value(scada, "USB2019_AI1")
```

```
57
58 M7026_AI0 := scada_tag_value(scada, "M7026_AI0")
59 M7026_AI1 := scada_tag_value(scada, "M7026_AI1")
60
61
62 if(scada_tag_event_value(scada, "M7026_AI0", "event"))
63 {
64     scada_tag_write_value(scada, "Indicator2", 1)
65 }
66 else
67     scada_tag_write_value(scada, "Indicator2", 0)
68 if(scada_tag_event_value(scada, "M7026_AI1", "event"))
69 {
70     scada_tag_write_value(scada, "Indicator3", 1)
71 }
72 else
73     scada_tag_write_value(scada, "Indicator3", 0)
74
75 }
76 }
77
78 on_event(t, scada_script())
79
```