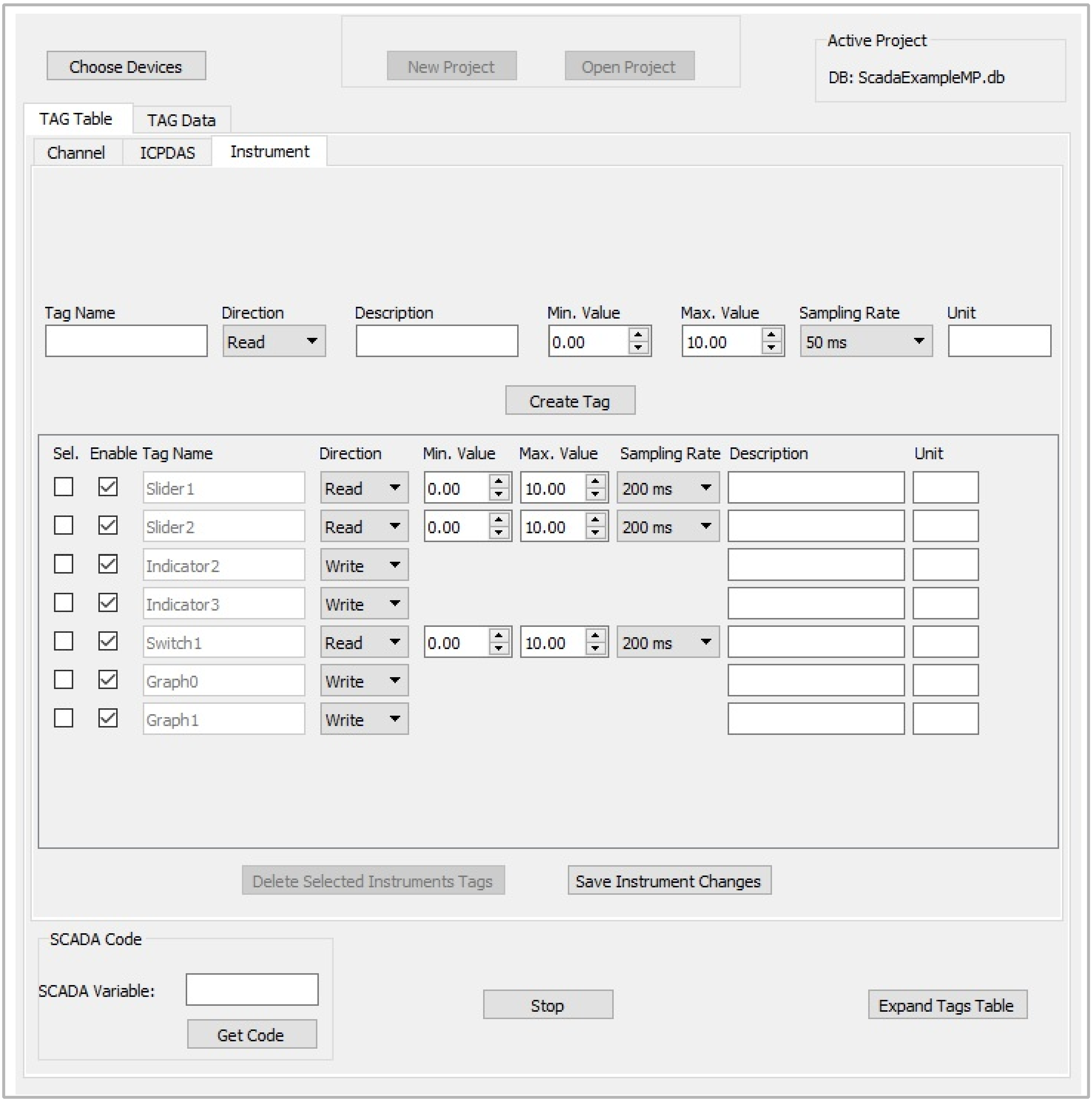
MatDeck Easy SCADA with ICP DAS Devices

MatDeck Easy SCADA gives users the ability to build industrial applications with ICP acquisition devices. 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 in a simple scenario using two ICP DAS devices, M-7026 and USB-2019, and virtual instruments provided by MatDeck.

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 which is used to build SCADA applications as illustrated below.

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



Active Project

DB: ScadaExampleMP.db

Tag Name	Direction	Description	Min. Value	Max. Value	Sampling Rate	Unit
	Read		0.00	10.00	50 ms	

Sel.	Enable	Tag Name	Direction	Min. Value	Max. Value	Sampling Rate	Description	Unit
☐	☑	Slider1	Read	0.00	10.00	200 ms		
☐	☑	Slider2	Read	0.00	10.00	200 ms		
☐	☑	Indicator2	Write					
☐	☑	Indicator3	Write					
☐	☑	Switch1	Read	0.00	10.00	200 ms		
☐	☑	Graph0	Write					
☐	☑	Graph1	Write					

SCADA Code

SCADA Variable:

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 AO, 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 show current values.

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

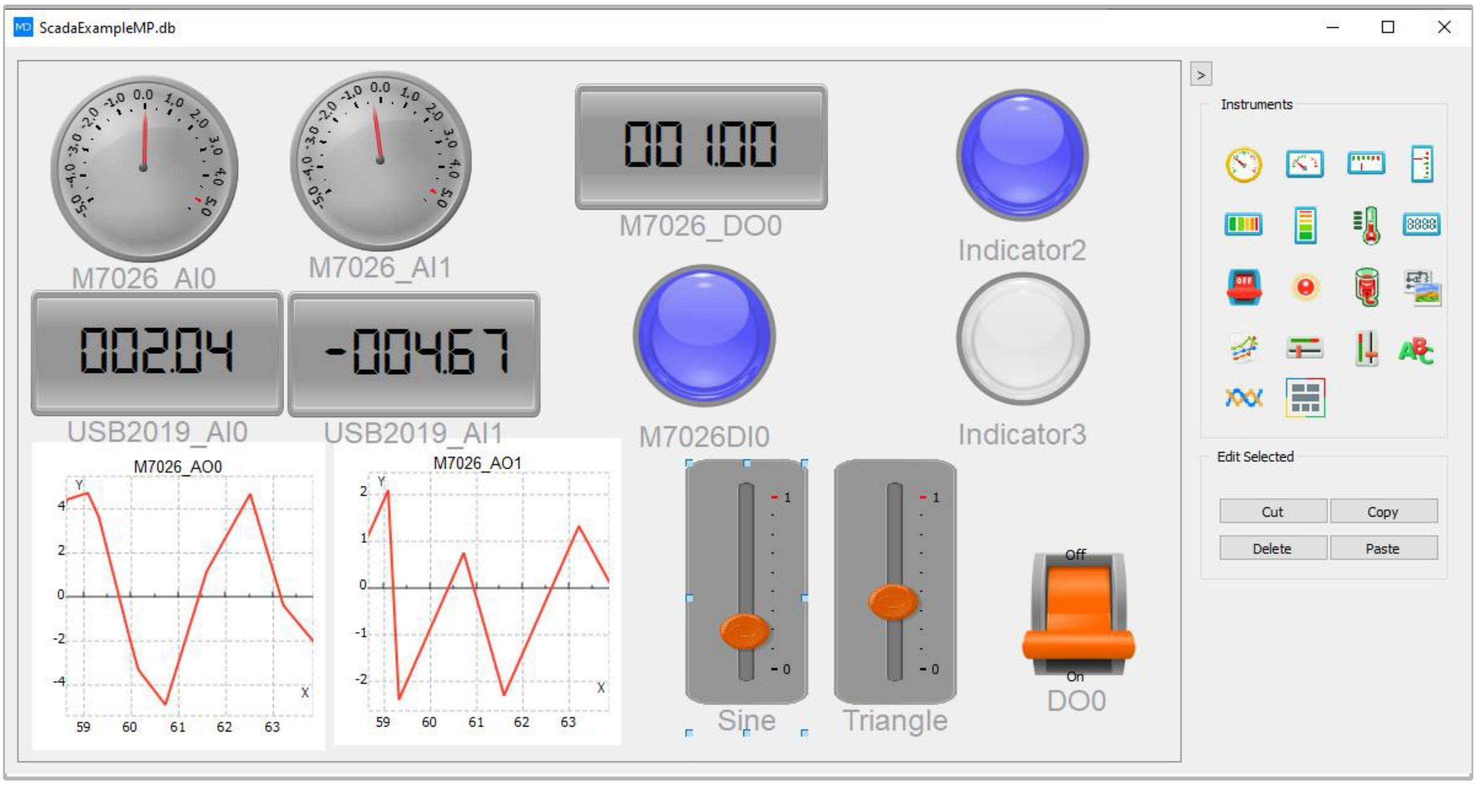
- If 2V<M-7026 A0I<3V Indicator2 ON
- If 0V< M-7026 AI1<3V Indicator3 ON

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

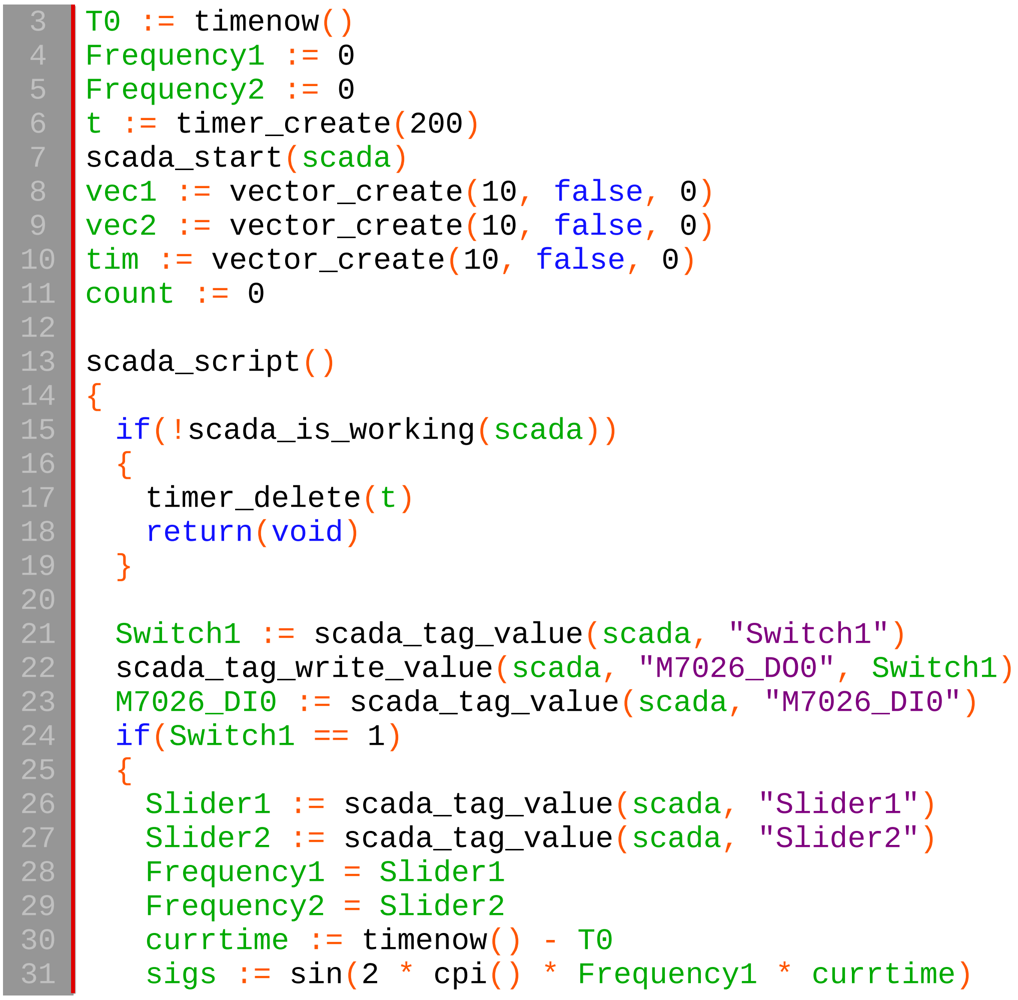
SCADA Panel

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



Script Code

The code which implements the SCADA application scenario is shown below.



```
T0 := timenow()
Frequency1 := 0
Frequency2 := 0
t := timer_create(200)
scada_start(scada)
vec1 := vector_create(10, false, 0)
vec2 := vector_create(10, false, 0)
tim := vector_create(10, false, 0)
count := 0

scada_script()
{
  if(!scada_is_working(scada))
  {
    timer_delete(t)
    return(void)
  }

  Switch1 := scada_tag_value(scada, "Switch1")
  scada_tag_write_value(scada, "M7026_DO0", Switch1)
  M7026_DI0 := scada_tag_value(scada, "M7026_DI0")
  if(Switch1 == 1)
  {
    Slider1 := scada_tag_value(scada, "Slider1")
    Slider2 := scada_tag_value(scada, "Slider2")
    Frequency1 = Slider1
    Frequency2 = Slider2
    currtime := timenow() - T0
    sigs := sin(2 * cpi() * Frequency1 * currtime)
```

```

   sawt := currtime * Frequency2 - floor(currtime * Frequency2 + 0.5)
   tim[count] = currtime
   vec1[count] = 5 * sigs
   vec2[count] = 5 * sawt
   //count += 1

   if(count < 9)
   {
         count += 1
   }
   else
   {
     gr1 := join_mat_cols(tim, vec1)
     gr2 := join_mat_cols(tim, vec2)
     scada_tag_write_value(scada, "Graph0", gr1)
     scada_tag_write_value(scada, "Graph1", gr2)
     count = 0
   }
   scada_tag_write_value(scada,"M7026_AO0", 5 * sigs)
   scada_tag_write_value(scada,"M7026_AO1", 5 * sawt)

   USB2019_AI0 := scada_tag_value(scada, "USB2019_AI0")
   USB2019_AI1 := scada_tag_value(scada, "USB2019_AI1")
   M7026_AI0 := scada_tag_value(scada, "M7026_AI0")
   M7026_AI1 := scada_tag_value(scada, "M7026_AI1")


   if(scada_tag_event_value(scada, "M7026_AI0", "event"))
   {
     scada_tag_write_value(scada, "Indicator2", 1)
   }
   else
     scada_tag_write_value(scada, "Indicator2", 0)
   if(scada_tag_event_value(scada, "M7026_AI1", "event"))
   {
     scada_tag_write_value(scada, "Indicator3", 1)
   }
   else
     scada_tag_write_value(scada, "Indicator3", 0)

  }
}

on_event(t, scada_script())

```