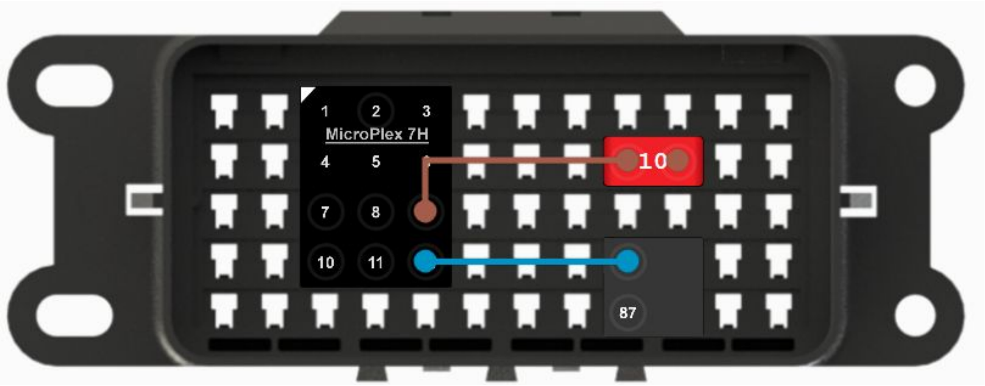
TD-138 MicroPlex 7H Sample Program Description

Introduction

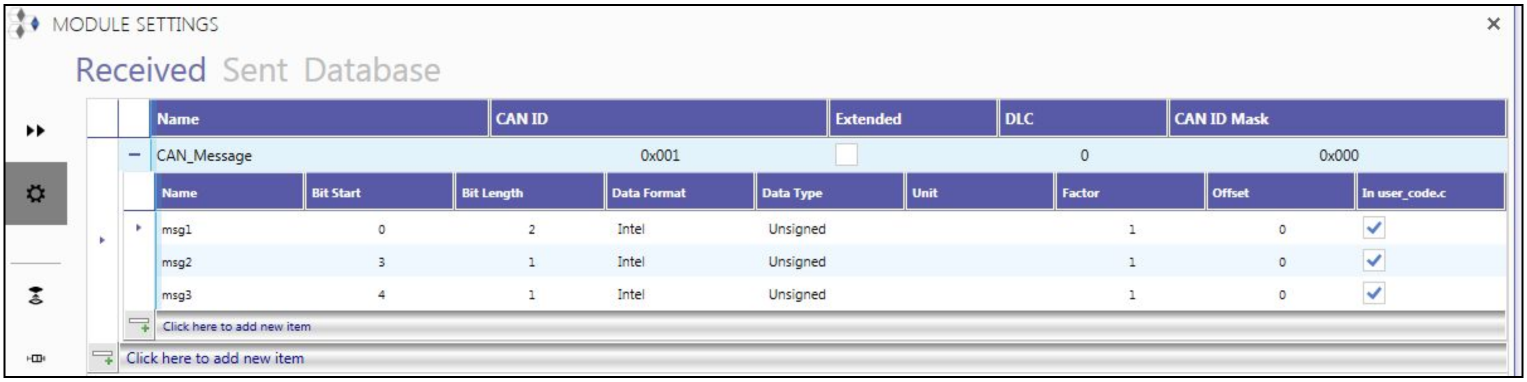
In this sample program for the MicroPlex 7H, we will walk through the set-up of the module settings in the MicroPlex Lab and explain what each part of the written code does. This program shows how to utilize the PWM feature for pin 12 while still using pin 9 as a basic digital output.

Component Layout and Module Settings

The following image shows the layout of the components; it is simply pin 12 on the MicroPlex 7H connected to an Ultra Micro relay and pin 9 connected to a 10 A fuse. To place the components just drag and drop them, and to place a wire just click on a module pin and drag to the other components.



Next, we must initialize the data in the Module Settings. This can be accessed by right clicking on the MicroPlex that has already been placed in the Lab. The settings used in the sample program are shown below.



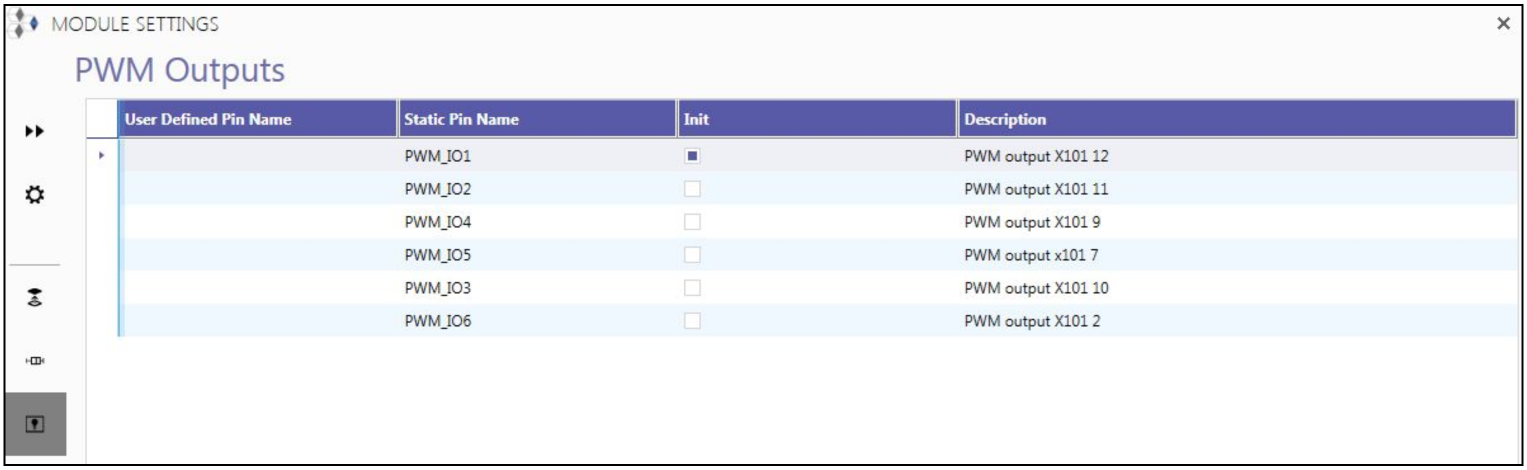
Name	CAN ID	Extended	DLC	CAN ID Mask
CAN_Message	0x001	□	0	0x000

Name	Bit Start	Bit Length	Data Format	Data Type	Unit	Factor	Offset	In user_code.c
msg1	0	2	Intel	Unsigned		1	0	☑
msg2	3	1	Intel	Unsigned		1	0	☑
msg3	4	1	Intel	Unsigned		1	0	☑

Click here to add new item

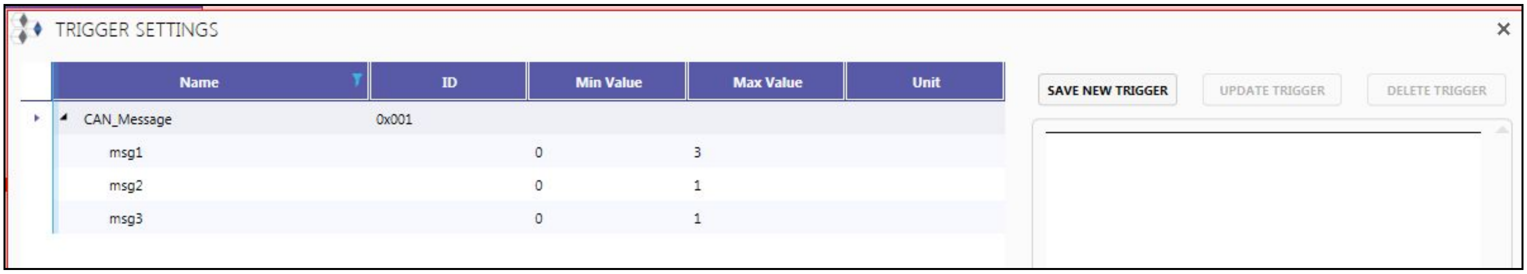
Click here to add new item

From this same window, click on the PWM tab on the left hand side and make sure that the PWM output for pin 12 is initialized:



User Defined Pin Name	Static Pin Name	Init	Description
	PWM_IO1	☑	PWM output X101 12
	PWM_IO2	□	PWM output X101 11
	PWM_IO4	□	PWM output X101 9
	PWM_IO5	□	PWM output x101 7
	PWM_IO3	□	PWM output X101 10
	PWM_IO6	□	PWM output X101 2

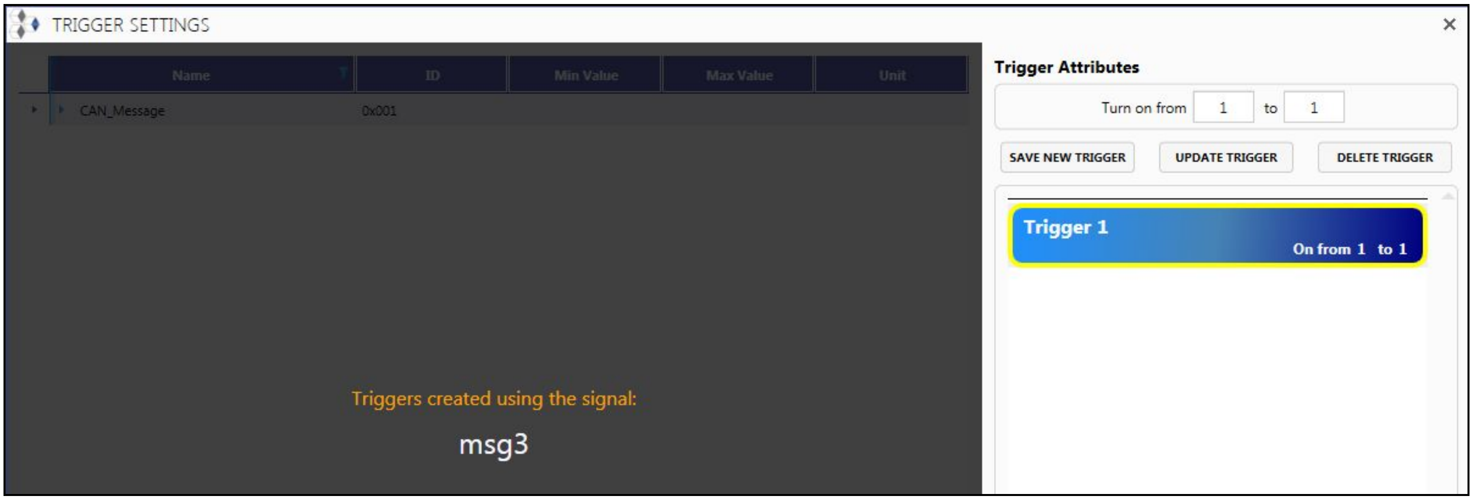
Once this is all setup, we can add a trigger to an output pin by double clicking on the wire that connects the MicroPlex to the relay. The following window will pop up once the wire has been double-clicked.



Name	ID	Min Value	Max Value	Unit
CAN_Message	0x001			
msg1		0	3	
msg2		0	1	
msg3		0	1	

By then selecting a data point, which we defined earlier in particular CAN messages from the Module Settings, we decide what controls this trigger. This means that triggers use predetermined bit(s) from a CAN bus message to control the output of a pin on the MicroPlex.

The sample program has a trigger assigned to pin 9 whose output now depends on the bit that was labeled as "msg3". The picture below shows what the window looks like after this trigger is set.



Name	ID	Min Value	Max Value	Unit
CAN_Message	0x001			

Triggers created using the signal:
msg3

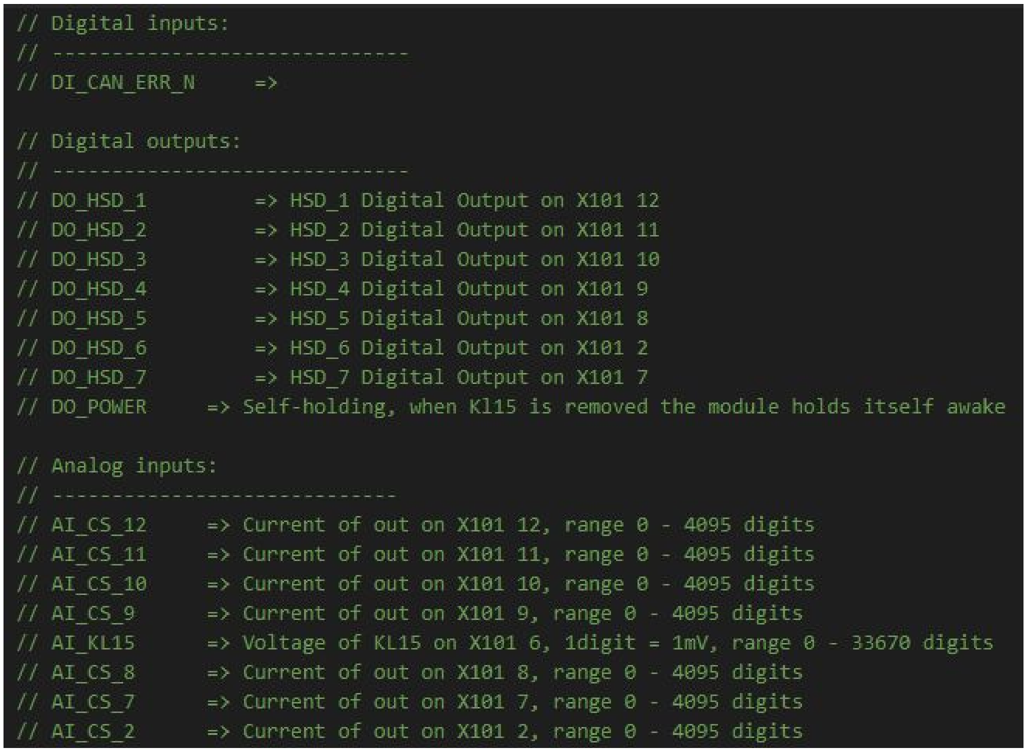
Trigger Attributes
Turn on from 1 to 1

Trigger 1
On from 1 to 1

Coding the MicroPlex Module

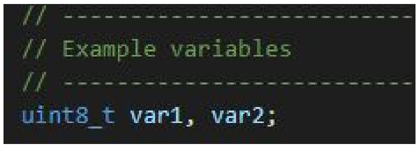
Now that we have finished designing the MicroPlex circuit and set up the module settings and triggers, we are ready to input code that is a little more complicated, but adds additional functionality. To open up the coding window, right click on the module and select "Open user_code.c".

There are many things that will already be written and commented out in this file. Much of this is an explanation of default functions and variables that the user can utilize. For instance, one of the first things you will see is a list of the names used to label certain outputs/inputs. A screenshot of this is found below. ('X101 12' is used to refer to pin 12 on the MicroPlex)



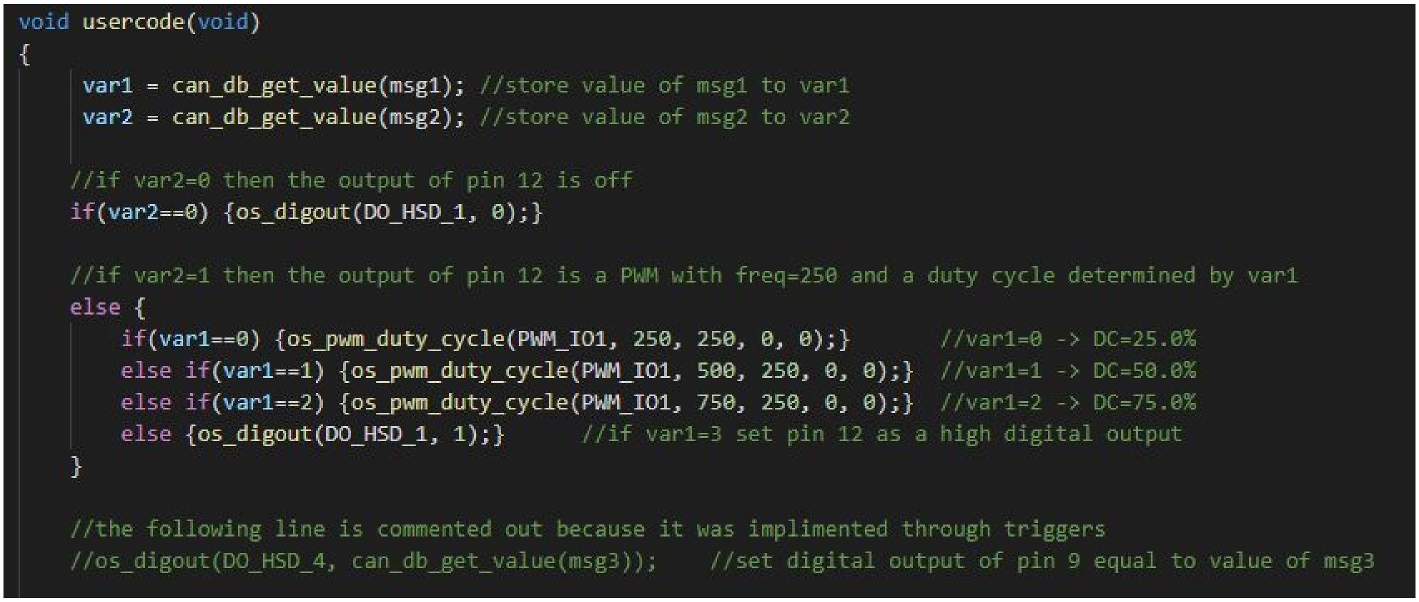
```
// Digital inputs:
// ------------------------------
// DI_CAN_ERR_N     =>

// Digital outputs:
// ------------------------------
// DO_HSD_1         => HSD_1 Digital Output on X101 12
// DO_HSD_2         => HSD_2 Digital Output on X101 11
// DO_HSD_3         => HSD_3 Digital Output on X101 10
// DO_HSD_4         => HSD_4 Digital Output on X101 9
// DO_HSD_5         => HSD_5 Digital Output on X101 8
// DO_HSD_6         => HSD_6 Digital Output on X101 2
// DO_HSD_7         => HSD_7 Digital Output on X101 7
// DO_POWER     => Self-holding, when Kl15 is removed the module holds itself awake

// Analog inputs:
// -----------------------------
// AI_CS_12     => Current of out on X101 12, range 0 - 4095 digits
// AI_CS_11     => Current of out on X101 11, range 0 - 4095 digits
// AI_CS_10     => Current of out on X101 10, range 0 - 4095 digits
// AI_CS_9      => Current of out on X101 9, range 0 - 4095 digits
// AI_KL15      => Voltage of KL15 on X101 6, 1digit = 1mV, range 0 - 33670 digits
// AI_CS_8      => Current of out on X101 8, range 0 - 4095 digits
// AI_CS_7      => Current of out on X101 7, range 0 - 4095 digits
// AI_CS_2      => Current of out on X101 2, range 0 - 4095 digits
```

Directly under the list of analog inputs should be some example variables. We want to change it to the following:



```
// --------------------------
// Example variables
// --------------------------
uint8_t var1, var2;
```

The majority of code we write goes in the function “void usercode(void){ }”. You can use “Ctrl+f” to search for this function within the code. This programs usercode is shown below.



```
void usercode(void)
{
     var1 = can_db_get_value(msg1); //store value of msg1 to var1
     var2 = can_db_get_value(msg2); //store value of msg2 to var2

    //if var2=0 then the output of pin 12 is off
    if(var2==0) {os_digout(DO_HSD_1, 0);}

    //if var2=1 then the output of pin 12 is a PWM with freq=250 and a duty cycle determined by var1
    else {
        if(var1==0) {os_pwm_duty_cycle(PWM_IO1, 250, 250, 0, 0);}       //var1=0 -> DC=25.0%
        else if(var1==1) {os_pwm_duty_cycle(PWM_IO1, 500, 250, 0, 0);}  //var1=1 -> DC=50.0%
        else if(var1==2) {os_pwm_duty_cycle(PWM_IO1, 750, 250, 0, 0);}  //var1=2 -> DC=75.0%
        else {os_digout(DO_HSD_1, 1);}      //if var1=3 set pin 12 as a high digital output
    }

    //the following line is commented out because it was implimented through triggers
    //os_digout(DO_HSD_4, can_db_get_value(msg3));    //set digital output of pin 9 equal to value of msg3
```

This code’s first two lines use the “can_db_get_value()” function to assign the values of msg1 and msg2 to var1 and var2. We then determine the PWM output of pin 12 with these variables and basic C commands.

The bottom of the code is a line that would just turn pin 9 on and off depending on msg3, but this has already been done with triggers previously, so I commented that line out.

As you can see, there are many ways to utilize the MicroPlex. If you have any questions or concerns regarding the product, please feel free to reach out to our team at Chief Enterprises!