# Local I/O

**Introduction**     This chapter provides information on how to create a new Project, connect to the PLC and download the Project to the PLC. This Project will include Local I/O.

Local I/O modules are on the same rack as the M580.

They are the most basic type of I/O and the easiest to configure.

To simplify I/O mapping most of the M580 I/O modules are configured via a Device DDT.

# Exercise - Configure Local I/O

**Learning Outcomes**

By the completion of this exercise the student will:

➢ Create a new M580 application

➢ Configure a local I/O

➢ Name and use a Device DDT variable

➢ Check the status of the local I/O drop

**Equipment Required**

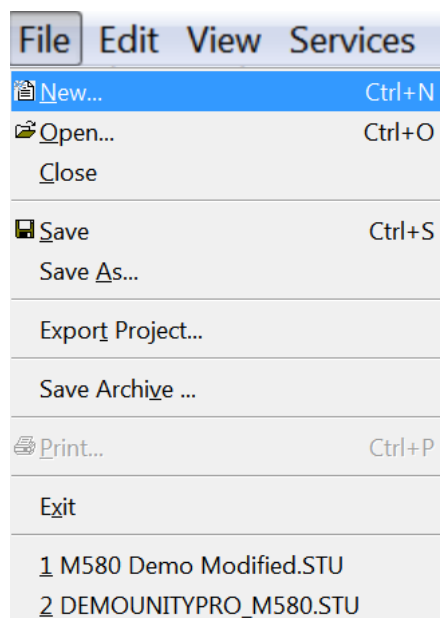To complete this exercise on a PLC the student will need

➢ One M580 PLC (any CPU)

➢ A compatible rack and power supply

➢ A DDO1602

## Create a new Project

i.  Using the Windows Start Menu open **Unity Pro**:

> **Start » All Programs » Schneider Electric » So Collaborative » Unity Pro » Unity Pro XL**

ii.  Create a **New Project** by selecting **File » New** from the Unity Pro menu, or clicking the **New Project** button on the toolbar.

| File | Edit View Services | |
|---|---|---|
| 🗎 New... | | Ctrl+N |
| 📂 Open... | | Ctrl+O |
| Close | | |
| 💾 Save | | Ctrl+S |
| Save As... | | |
| Export Project... | | |
| Save Archive ... | | |
| 🖶 Print... | | Ctrl+P |
| Exit | | |
| 1 M580 Demo Modified.STU | | |
| 2 DEMOUNITYPRO_M580.STU | | |

# Exercise - Configure a Local I/O (cont.)

iii.     Select the appropriate **M580 Processor** and **Rack** according to the
         equipment available. Or select any option if there isn't an **M580 Processor**
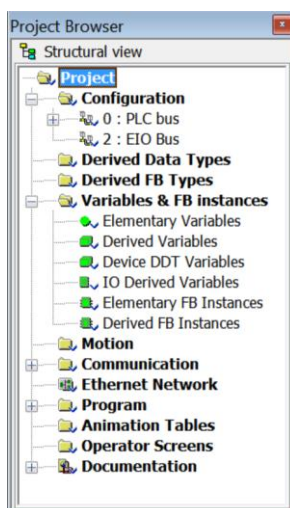         or **Rack** available.
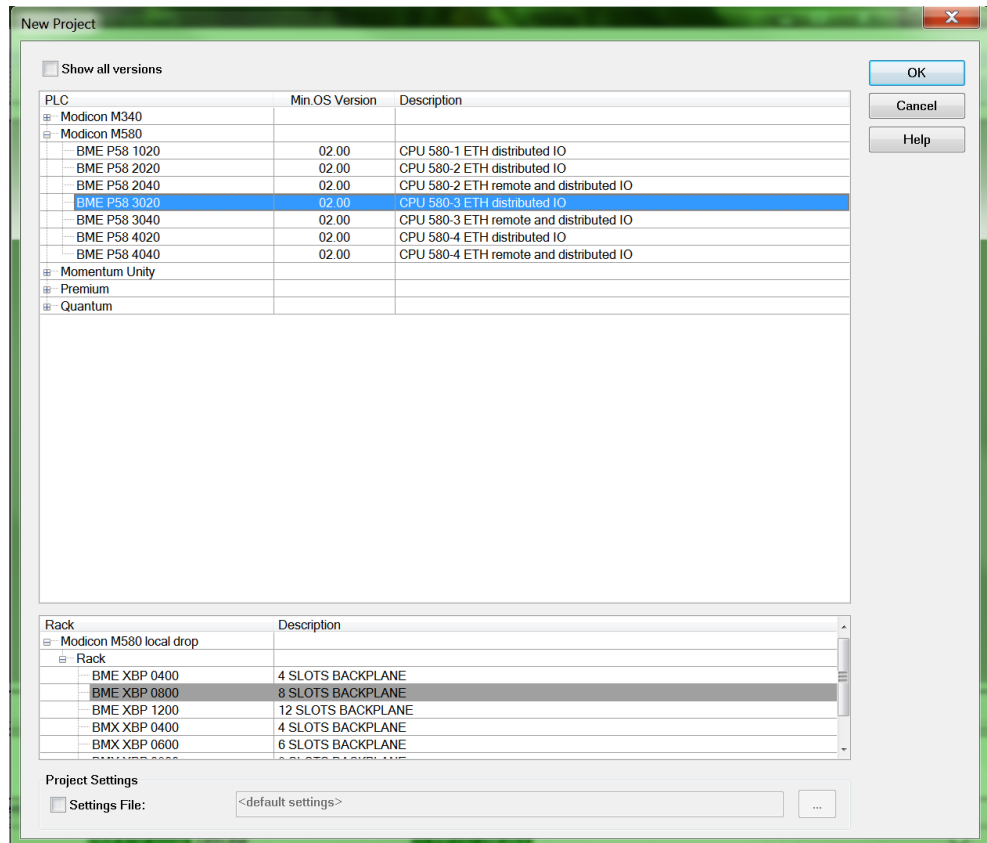
# Exercise - Configure a Local I/O (cont.)

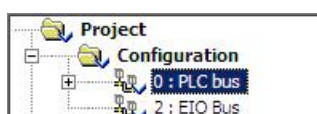iv. Click the **OK** button to create the application.

> **Unity Pro** will create the new project and populate it with default items.
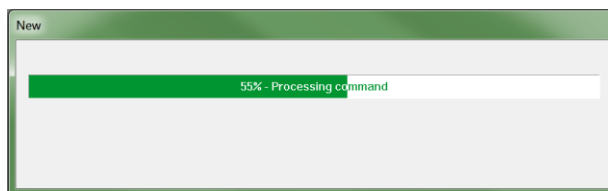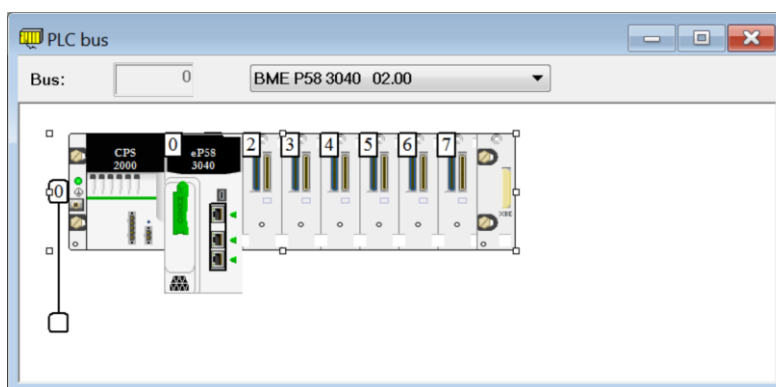


> The **Project Browser** will display to show the project contents.



v. Double-click the **0: PLC Bus** item from the **Project Browser.**



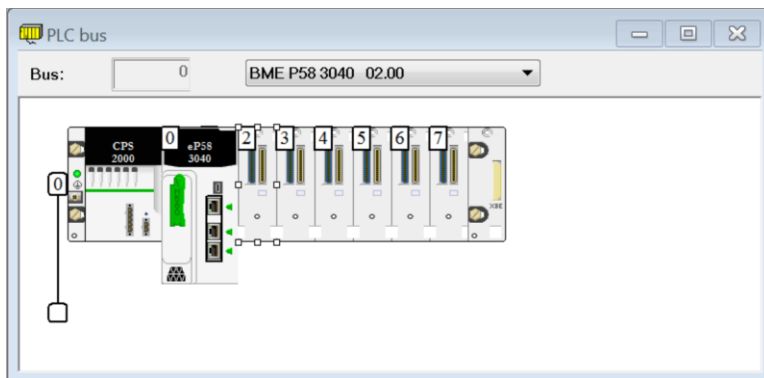> The Local Rack will be displayed, pre-populated with the CPU and the Power Supply.



---

💡 **Hints & Tips**

Note the addition of the new Ethernet slot in Blue on the image of the Rack.
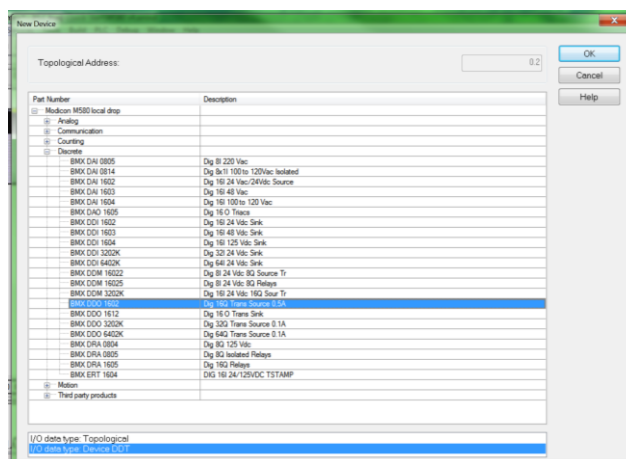
# Exercise - Configure a Local I/O (cont.)

**Add the DDO Module to the Local Rack.**

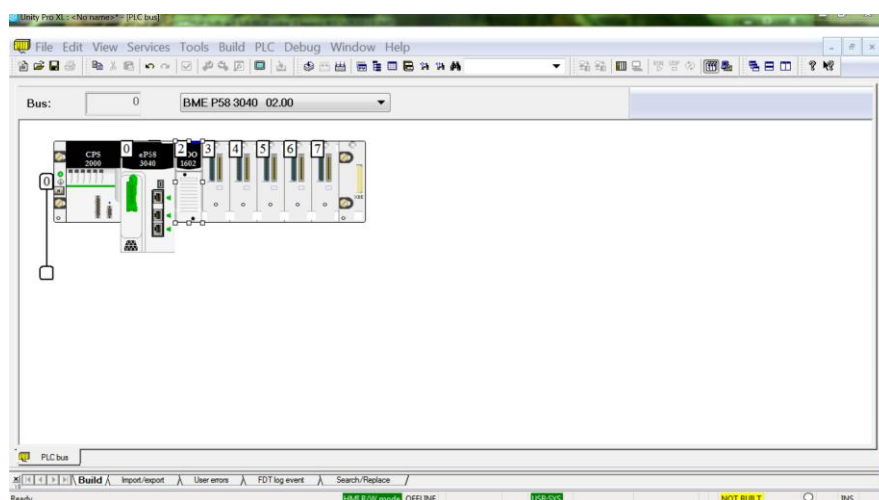i.    Double-click the spare slot representing the physical location of the **BMX DDO 1602** module.



ii.   From the **New Device** window, select the **Discrete** group and then select the **BMX DDO 1602** module.  Make sure that the **I/O data type is Device DDT** at the bottom.
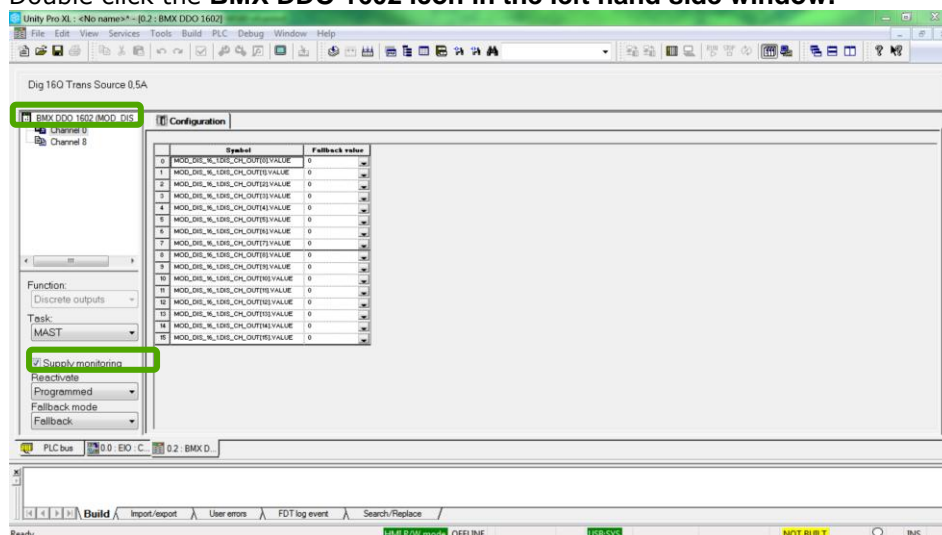
iii.  Click the **OK** button.

# Exercise - Configure a Local I/O (cont.)

The module appears in the Local Rack.



### Name the DDO 1602 Device DDT

i.   Double-click the **BMX DDO 1602** module.

ii.  In the module's window, un-tick **Supply monitoring**.

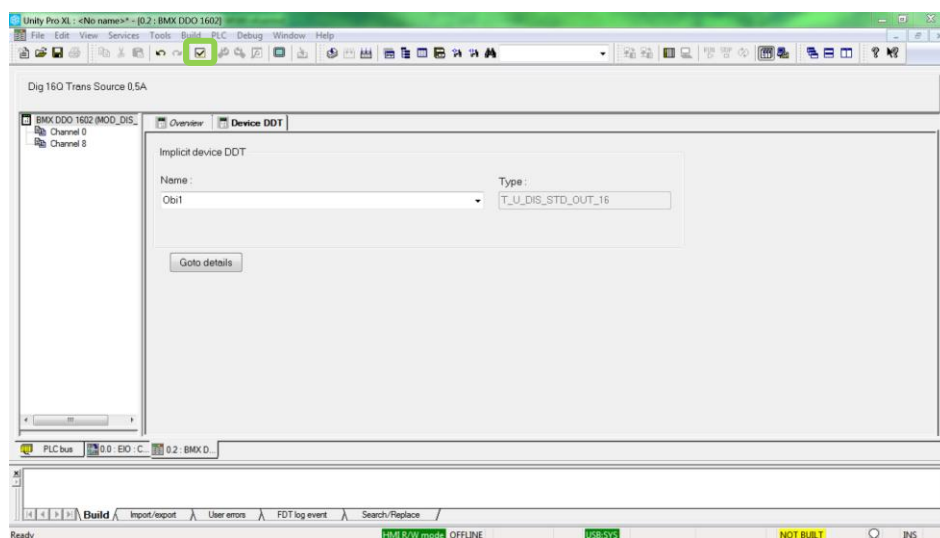iii. Double click the **BMX DDO 1602 icon in the left hand side window.**



✎ **Note:**
Choose any name, it can be changed anytime.

# Exercise - Configure a Local I/O (cont.)

iv.    Select the **Device DDT** Tab, rename the module `Obi1`.

v.    Validate the changes, by clicking the tick box.



### Create a Test Section

i.    Create a new ST Section under the Master (MAST) Task called **test**.

ii.    Type in the following code:

```
FOR i:=0 TO 15 BY 2 DO
    Obi1.DIS_CH_OUT[i].VALUE := TRUE;
END_FOR;
```

(create the variable i as an Integer)

iii.    This code will turn on every even output.

iv.    Build the application.

---

**Note:** The syntax of device DDT outputs.

---

# Exercise - Configure a Local I/O (cont.)

**Observe the DDO 1602 Device DDT**

   i.    Transfer and run the application (either to Simulation mode or Standard mode if the equipment is available).

   ii.    In the project browser, double click **Variables & FB instances**.

   iii.    Right click **Obi1**, and initialise a new animation table.

   iv.    Click the **+** to extend the structure.

   v.    Extend the **DIS_CH_OUT** item.

   vi.    Finally extend a few channels and check their states.

   vii.    Odd numbers should be OFF, and even numbers should be ON, as on the picture:

| Name | Value | Type | Comment |
|---|---|---|---|
| ⊟ Obi1 | | T_U_DIS_STD... | |
| MOD_HEALTH | 0 | BOOL | Module health |
| MOD_FLT | 0 | BYTE | Module faults |
| ⊟ DIS_CH_OUT | | ARRAY[0..15] O... | |
| ⊟ DIS_CH_OUT[0] | | T_U_DIS_STD... | |
| CH_HEALTH | 0 | BOOL | Channel health |
| VALUE | 1 | EBOOL | Discrete output value |
| ⊟ DIS_CH_OUT[1] | | T_U_DIS_STD... | |
| CH_HEALTH | 0 | BOOL | Channel health |
| VALUE | 0 | EBOOL | Discrete output value |
| ⊟ DIS_CH_OUT[2] | | T_U_DIS_STD... | |
| CH_HEALTH | 0 | BOOL | Channel health |
| VALUE | 1 | EBOOL | Discrete output value |

   viii.    Save the project.

# Exercise - Configure a Local I/O (cont.)

**Check the DDO 1602 Device Outputs (Hardware Required)**

i.  The Hardware described at the beginning of the exercise is required to complete this section.

ii.  Check that every other output is ON as in the picture:



🖎 **Note:**

If I/O is red, it probably means that Supply monitoring in "3 name your DDDT" is unchecked.