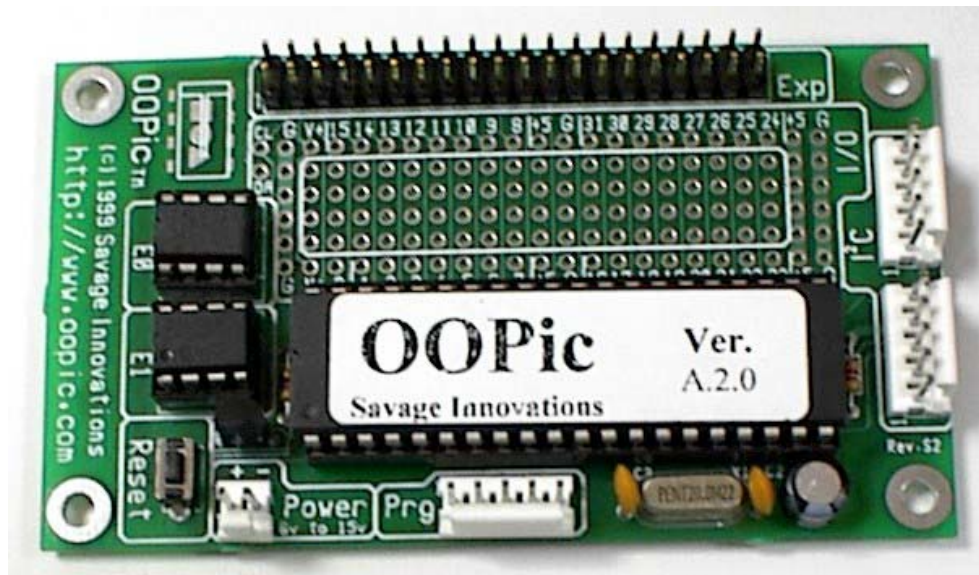


How To Program an OOPic I To Store Data In A Second EEPROM

By Abraham L. Howell

To get started you will need several components: your OOPic I controller, programming cable, and a suitable EEPROM chip that is to be properly placed in the E1 slot on your OOPic I. For this tutorial we will be using a 24LC64 EEPROM chip manufactured by [Microchip](#). This little EEPROM chip is capable of storing up to 8K x 8 (64K bits) of data. The OOPic I is shipped with a 24LC32 EEPROM chip installed in the E0 slot, which is used to store your downloaded program, which can be up to 4K x 8 (32K bits) in size. To install the 24LC64 into the E1 slot, simply insert the chip with the same orientation as the 24LC32 chip that is already installed in the E0 slot. Please refer to the picture below for an accurate installation. There should be a small notch on one end of the 24LC64 chip and this notch should facing away from the larger chip, which has the OOPic label attached to it.



Now that we have the 24LC64 properly installed, we can get down to the business of writing some code to save and then retrieve data from it. We will be using the oI2C object to access the 24LC64 chip. First, we must declare an instance of our oI2C object and give it a name. I will be naming my oI2C object E1, since the chip is installed in the E1 slot on the OOPic, but you may call it whatever you like. We will declare our E1 object as shown below.

Dim E1 as New oI2C

We must also properly configure our E1 oI2C object before we may access it in the main part of our program. I usually like to write a small initialization routine in which all object initialization is taken care of before the program actually executes the Main Sub routine in my program. The first thing we need to do is to configure the correct node for our E1 object.

```
Sub Initialize ( )  
    E1.Node=cvE1  
End Sub
```

Additionally, we must decide whether we will be storing bytes or words of data. In this example we will be storing bytes of data and therefore must set the Width property of our E1 object to be cv8Bit. If we were going to be storing words then we would simply set the width property to cv16Bit.

```
Sub Initialize ( )  
    E1.Node=cvE1  
    E1.Width=cv8Bit  
End Sub
```

Next we must set the starting location for our E1 object to be 1, since we want to start writing data at the beginning of memory in our 24LC64 chip.

```
Sub Initialize ( )  
    E1.Node=cvE1  
    E1.Width=cv8Bit  
    E1.Location=1  
End Sub
```

We would also like it if our E1 object would automatically increment its Location property by 8Bits every time we write data to it. To do this we simply set the NoInc property of our E1 object to cvFalse. With the NoInc property set to cvFalse the Location property will automatically be increased by whatever we have the Width property set to and in this case it will be incremented by 8Bits.

```
Sub Initialize ( )  
    E1.Node=cvE1  
    E1.Width=cv8Bit  
    E1.Location=1  
    E1.NoInc=cvFalse  
End Sub
```

We are now ready to begin using our E1 object in the Main Sub of our program. Lets save the following bytes of data to the 24LC64 using our E1 object: 225, 165, 10, and 29.

```

Sub Main ()
    Initialize
    E1.Value=225
    E1.Value=165
    E1.Value=10
    E1.Value=29
End Sub

```

What if we wanted to retrieve a specific piece of data at say memory location 3 and store it in an oByte object called E1_Data? Well we would simply set the E1 Location property to 3 and then read the data into our E1_Data oByte object as shown below. We would however need to declare an instance of this E1_Data object before using it.

```

Dim E1 as New oI2C
Dim E1_Data as New oByte

Sub Main ()
    Initialize
    E1.Value=225
    E1.Value=165
    E1.Value=10
    E1.Value=29
    E1.Location=3
    E1_Data.Value=E1.Value
End Sub

```

Suppose we wanted to save bytes and words of data? Well we would simply have to remember to adjust the Width property of our E1 oI2C object before writing any data to it. Storing bytes and words also adds another problem to the mix, since now we must remember where we have stored bytes and where we have stored words, otherwise we will not be able to accurately retrieve them later on. Say initially we start with a byte in Location 1 and then store a word, then again a byte, and then a word and continue this pattern of data storage. Now we want to go back and retrieve the second word that was stored. The easiest way to do this would be to set the Width property to cv16Bit and then set the Location property to 3. Now we would be able to read whatever word size piece of data we previously stored. This seems counter-intuitive, but we must remember our data storage scheme: byte, word, byte, word, byte, word... etc. So to access the second word stored we would have to skip over a total of 4bytes or 2words. That is why we decided to set the Width property to cv16Bit and then index to Location 3. For more information with regards to the OOPic line of Microcontrollers, please visit the OOPic website at <http://www.oopic.com>

If you have any questions about this tutorial, please send an email to abe@abotics.com

Click [here](#) to download the source code for this tutorial.