

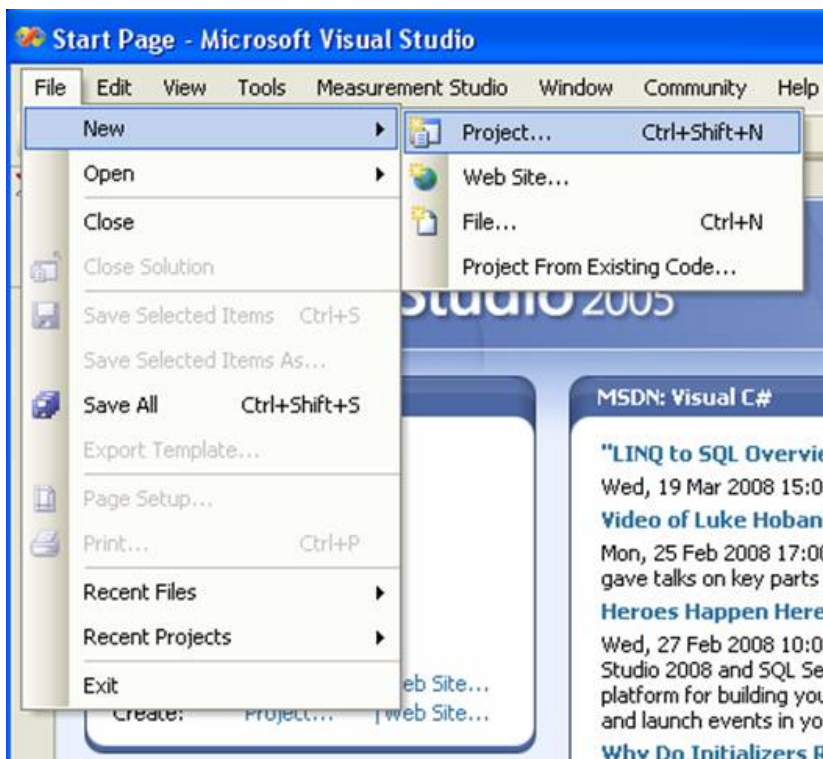
# Using Existing C Code or a DLL in LabVIEW

---

This tutorial explains how to convert existing C code to a DLL (Dynamically Linked Library) and then how to call that DLL from LabVIEW. If you already have a DLL, start on step 10. (Note: This tutorial explains how to create a DLL that can only be used in LabVIEW. If you want to create a DLL out of your C code that will then be downloaded to the FIRST cRIO controller, see this [article](#).)

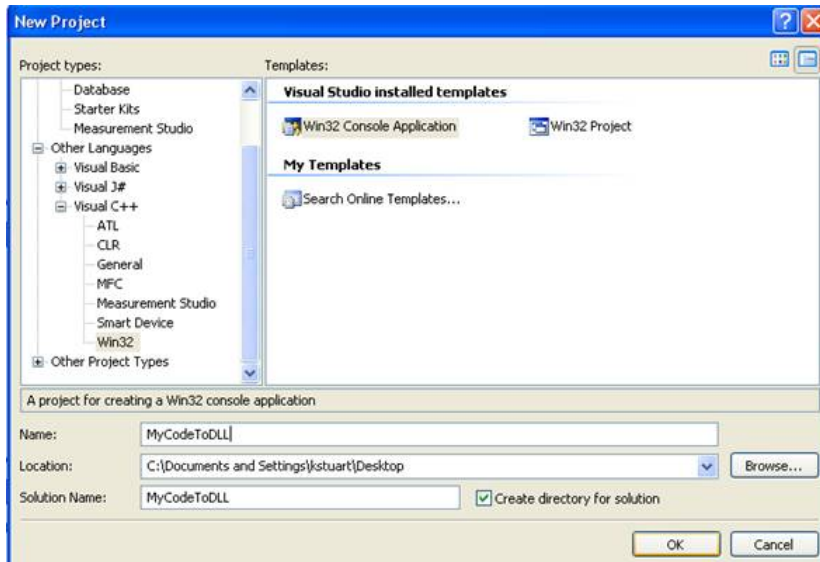
1. Use an integrated development environment capable of creating DLLs. (This tutorial uses Microsoft Visual Studio 2005.)

2. Create a new project.



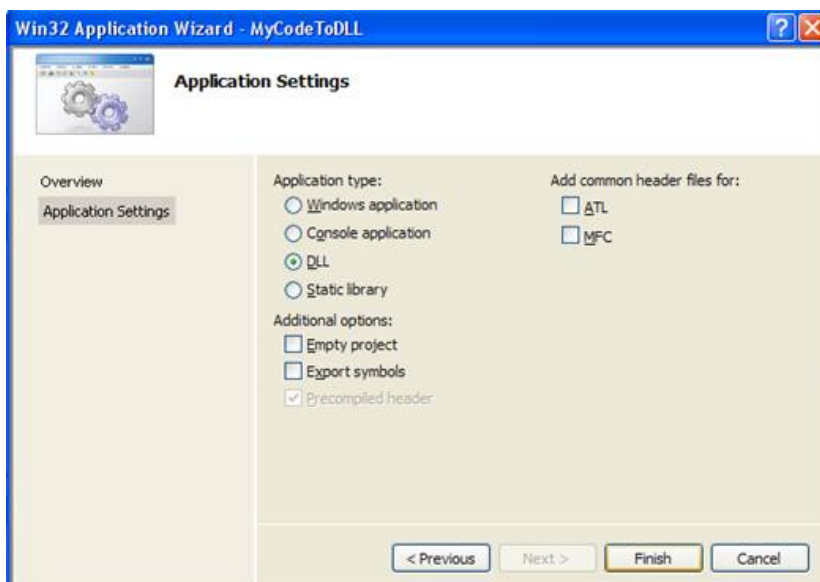
3. Create a C++ Win32 Console Application. Specify a Name. Specify a Location.

## Using Existing C Code or a DLL in LabVIEW



4. Click Next for more options.

5. Select DLL from Application type.



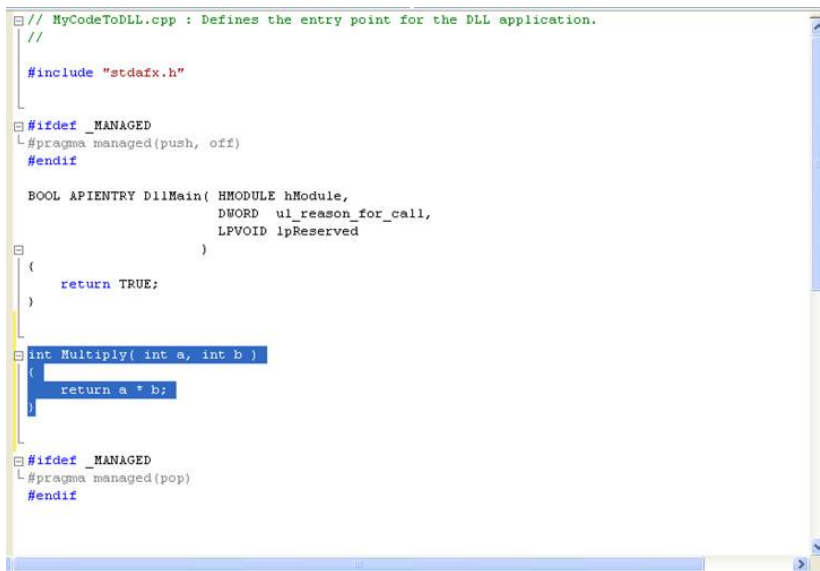
6. Add your function to the default program. (See my function in bold.)

```
int Multiply( int a, int b )
```

```
{
```

```
return a*b;
```

```
}
```



```
// MyCodeToDLL.cpp : Defines the entry point for the DLL application.
//
#include "stdafx.h"

#ifdef _MANAGED
#pragma managed(push, off)
#endif

BOOL APIENTRY DllMain( HMODULE hModule,
                      DWORD  ul_reason_for_call,
                      LPVOID lpReserved
                      )
{
    return TRUE;
}

int Multiply( int a, int b )
{
    return a * b;
}

#ifdef _MANAGED
#pragma managed(pop)
#endif
```

7. Add the following prototype to the default program. (See my prototype in bold.)

```
extern "C" __declspec(dllexport)int Multiply( int a, int b );
```

## Using Existing C Code or a DLL in LabVIEW

```
// MyCodeToDLL.cpp : Defines the entry point for the DLL application.
//
#include "stdafx.h"

#ifdef _MANAGED
#pragma managed(push, off)
#endif

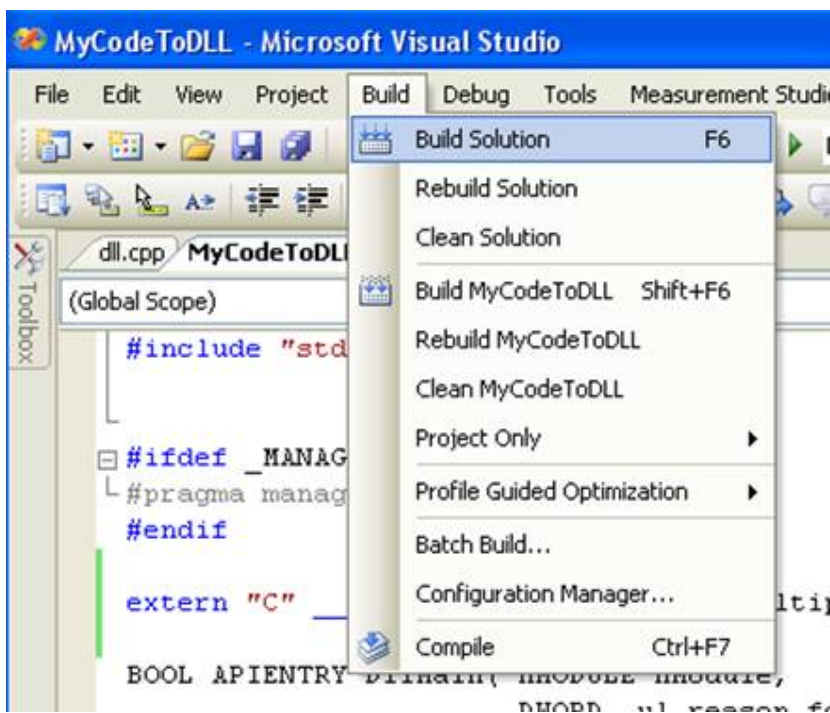
extern "C" __declspec(dllexport) int Multiply( int a, int b );

BOOL APIENTRY DllMain( HMODULE hModule,
                      DWORD  ul_reason_for_call,
                      LPVOID lpReserved
                      )
{
    return TRUE;
}

int Multiply( int a, int b )
{
    return a * b;
}

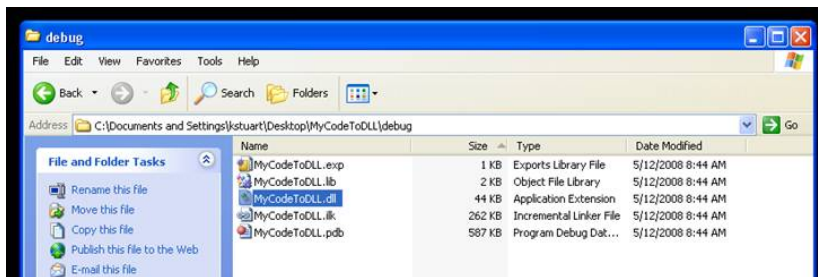
#ifdef _MANAGED
#pragma managed(pop)
#endif
```

### 8. Build the DLL.

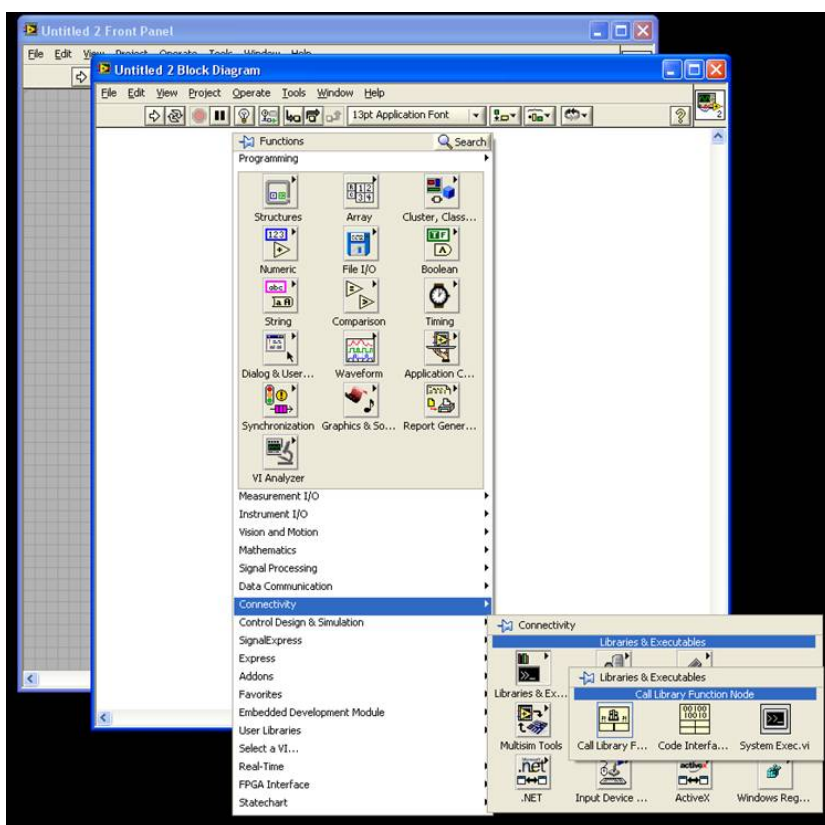


### 9. Note where your DLL was created.

## Using Existing C Code or a DLL in LabVIEW

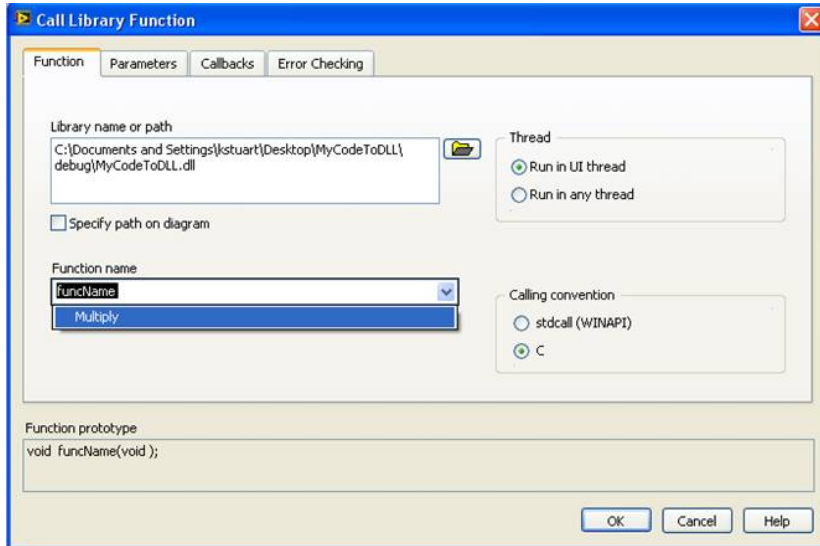


10. Open LabVIEW and place a Call Library Function Node on the Block Diagram.



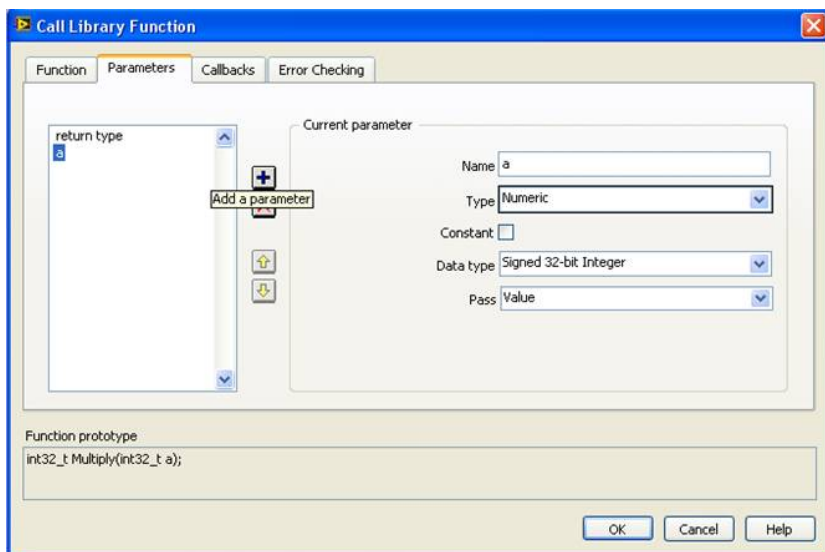
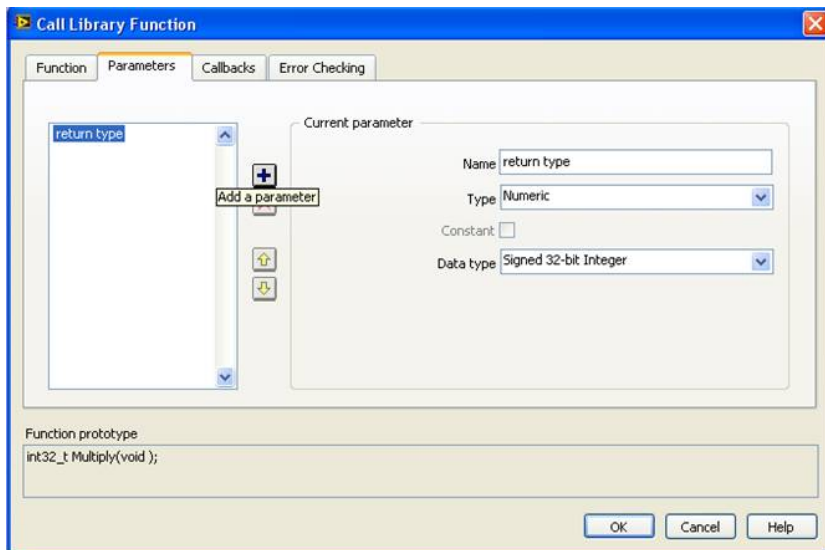
11. Double click on the Node and specify the DLL to load and the function to access.

## Using Existing C Code or a DLL in LabVIEW

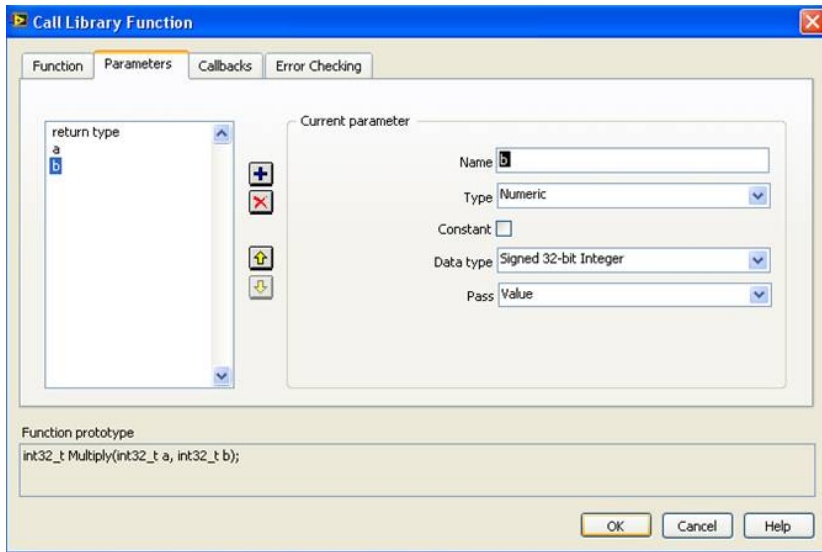


12. Under the Parameters tab, specify the Return Type (the data being passed back from the function) and press Add a parameter in order to specify each of your input parameters.

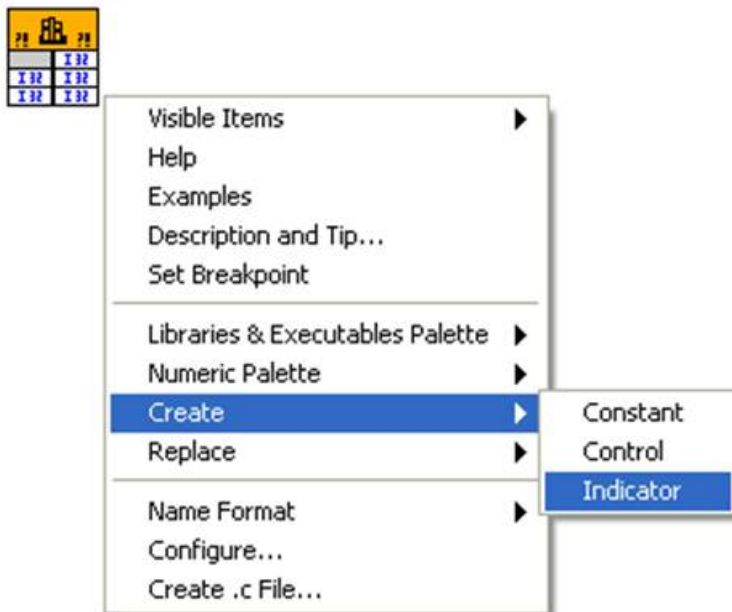
## Using Existing C Code or a DLL in LabVIEW



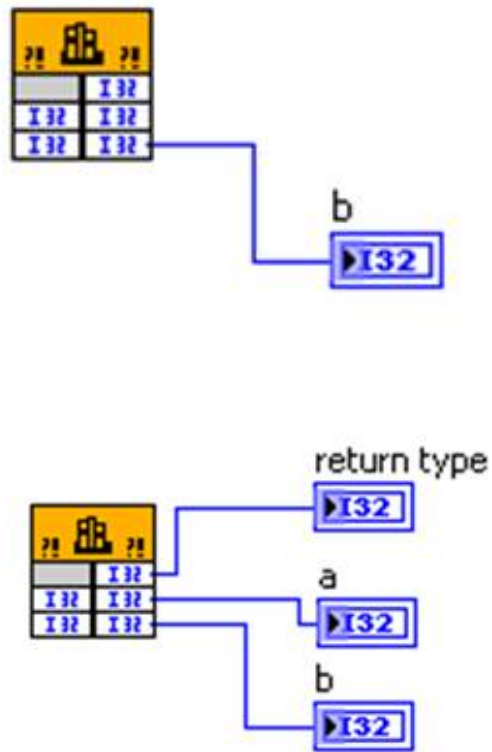
## Using Existing C Code or a DLL in LabVIEW



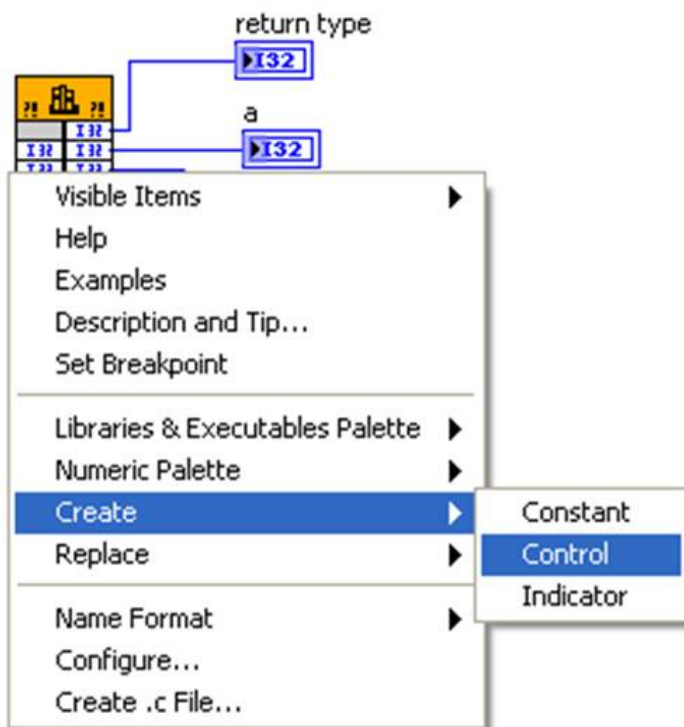
13. Right click on each of the 3 outputs and create an indicator. (Note: You have the option of viewing as outputs each of the input parameters to the DLL.)

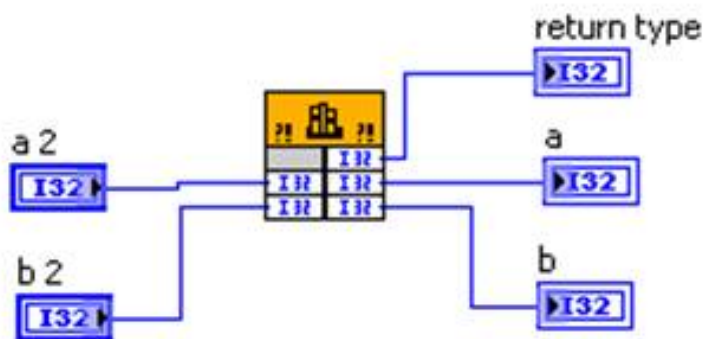
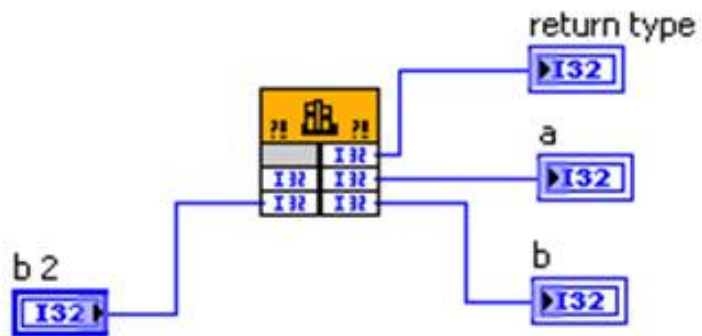






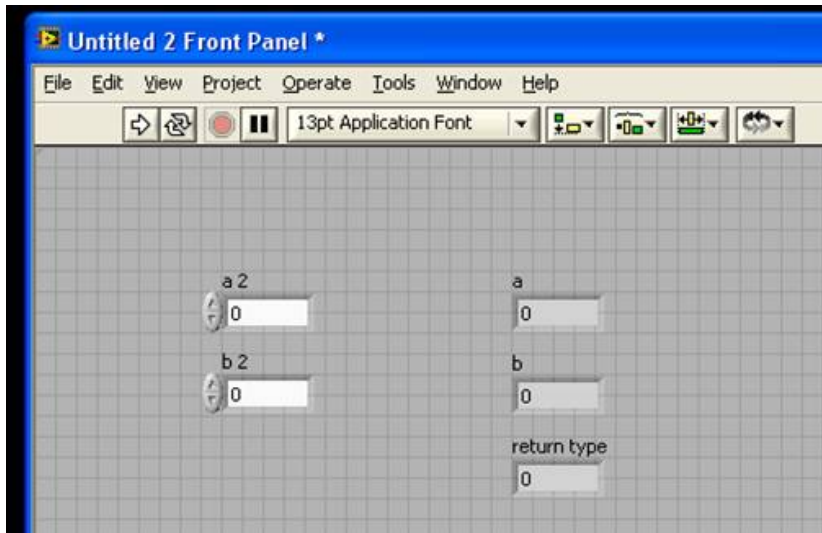
14. Right click on each of the 2 inputs and create controls.





15. Go to the Front Panel. Specify values for the 2 controls (the input parameters).

## Using Existing C Code or a DLL in LabVIEW



16. Run the VI in order to execute the DLL.

