New Tools from Microchip!

PICkit 5 and MPLAB ICD 5 Available Now!

By Tam Hanna, for Microchip

Long gone are the days when electronics enthusiasts cobbled together homebrew burners to program PIC and similar microcontrollers. Microchip has recently introduced their latest PICkit 5 In-Circuit Debugger/Programmer which won't break the bank and is compatible with many of the Microchip controller families. For professional troubleshooting, the powerful MPLAB ICD 5 In-Circuit Debugger/Programmer is also available. This article will explore their new features and demonstrate how to integrate the new versions into an existing development ecosystem.

Now with USB-C!

Microchip has always been committed to reducing the "cable clutter" in labs of MPLAB developers. The earlier ICD 3 variant used a USB-A connector, while the PICkit 4 came with a Micro-USB connector.

Standardization in the USB connector world now means that both PICkit 5 [1] and the ICD 5 [2] come with a USB-C connector, which is becoming the standard for all modern smartphones. Universal uptake will eventually make USB cable variant anxiety a thing of the past.

For owners of these devices, this comes with several advantages. Firstly, for a developer, it's now easy to use the same cable you use to recharge your smartphone to connect to your development environment. This saves space — and sometimes money — on business trips because you can borrow USB-C cables almost anywhere due to their widespread use in the smartphone industry.

The second innovation specifically concerns owners of the "larger" programming device variant, the ICD 5. Due to its access to more power, the ICD 5 can provide up to 1 A to the application circuit. In many cases, this eliminates the need to carry a second power supply for the device under test. The smaller device is still limited to a current budget of around 150 mA — likely a concession to its significantly smaller hardware.

Regarding actual programming speed, there are no significant advantages with USB-C — both programmers still operate at "only" the USB 2.0 High Speed bitrate, but this should not be a major issue in practice, given the small file size of most microcontroller images. Hardware optimizations, on the other hand, enhance the "perceived speed."

And PoE!

When developing applications that include high voltage sections in the design, it's important to ensure good electrical separation between the high voltage stage and the rest of the circuit, including the processor, low-voltage signals, and sensor signals. One advantage of using a communication link such as an Ethernet network is that it usually uses isolation transformers at each network socket to provide galvanic isolation of the data signals. This provides good isolation between your expensive workstation or PC and any external device.

The latest ICD 5 includes a network port on the back (**Figure 1**). What's new is that it can now supply power to the "entire application" through Power over Ethernet (PoE) if configured appropriately in your PoE network. This can provide substantial current to the application circuit and means you won't need to provide an additional mains adapter to power it.

Fortunately, MPLAB X is also ready to communicate directly with the ICD 5 via the internet, so there's no need to wire an additional USB-C cable between your PC and the programming device.

It's worth noting that this setup also encourages experimentation with Continuous Integration and Continuous Delivery, although older versions of the programming device offered similar functionality when used with an Ethernet connection and a 9 V mains adapter.

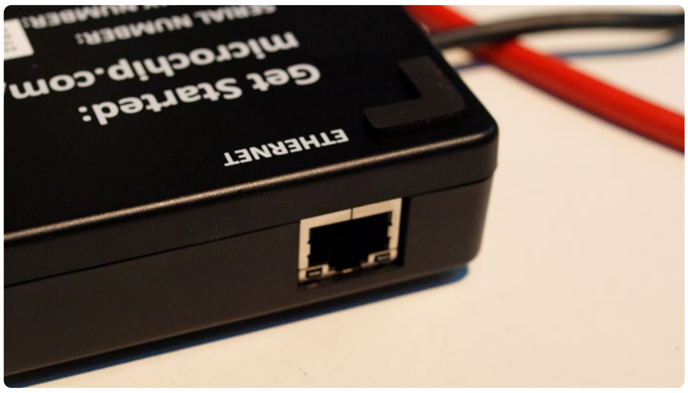


Figure 1: This port also provides Power over Ethernet capability.

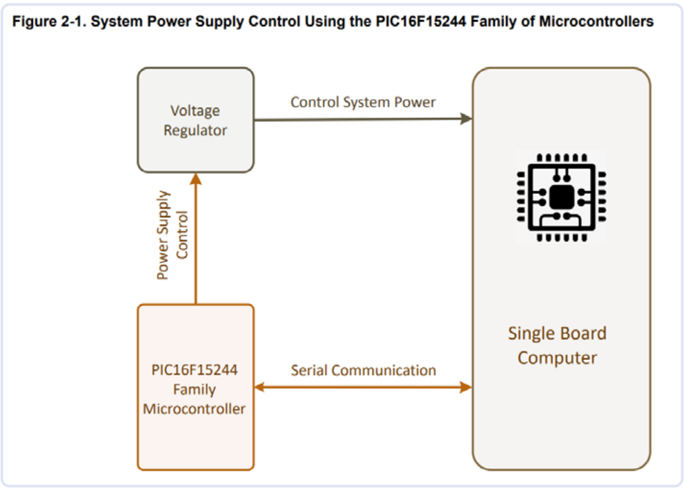


Figure 2-1. System Power Supply Control Using the PIC16F15244 Family of Microcontrollers
Voltage Regulator
Control System Power
Power Supply Control
Single Board Computer
PIC16F15244 Family Microcontroller
Serial Communication

Figure 2: Even a management microcontroller requires firmware! (Source: [3])

The wall wart PSU supplied with earlier versions of the development kit is no longer part of the package in Version 5. This indicates that power and also be acquired through the higher power capability that USB-C or USB 3.0 ports have when compared to earlier USB incarnations.

PICkit 5: Improved Functionality

The actual development of an electronic system is often just the first part of a complex value chain. Those who don't consider "Design for Manufacturing" — and similar aspects — from the start may encounter unexpectedly high costs or problems during manufacturing.

A good example is systems that combine a "management microcontroller" with a process computer, as described in Application Note AN4121 [3] and schematically summarized in **Figure 2**. Failing to plan properly, such as poorly positioning the microcontroller programming port, can significantly impact assembly and product maintenance costs.

While developers can handle delivering firmware to each individual board of a prototype system using MPLAB, this method will become time-consuming and unworkable when the device enters production.

With the PICkit 5 and its *Programmer-To-Go* feature, Microchip offers a solution. Essentially, this feature has been available for several versions, and the secret lies in the microSD slot shown in **Figure 3**, which accommodates a FAT32-formatted memory card.

Using MPLAB, developers can define a firmware image to be deployed automatically, and the programming adapter can then deliver it to connected target devices with minimal effort.

What's also new with the PICkit 5 is the ability to copy more than one firmware image into the *Programmer-To-Go* storage area. This allows a programming device to be used for the "parameterization or final assembly" of multiple assemblies. Initiation of the programming process occurs via the familiar button hidden beneath the logo on the front of the unit, so image selection must be done via a different communication interface.

...and Friends to the Bluetooth Radio Standard

To set up the new programming devices, you'll need development environment MPLAB version 6.10 or higher — the actual installation process is pretty much standard, as you'd expect from MPLAB. Besides

Figure 3: The memory card slot.

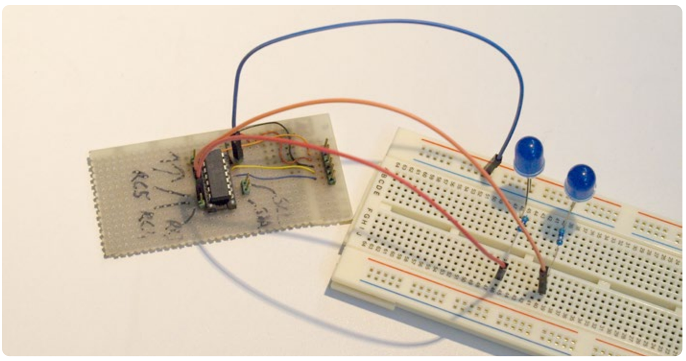


Figure 4: This old board will serve as a test system.

the typical installation of the XC8 compiler, which Microchip traditionally handles separately, make sure that, during the initial startup of MPLAB, you grant it access to both local and remote networks in Windows Firewall. This is especially important if you plan to use the larger version of the programming device with its Ethernet connection.

For the next step, you'll need an SD card. I used a spare 16 GB card and formatted it with a FAT32 partition under Ubuntu. Look for a special feature of the Disk Management snap-in; one quirk of the app is that it doesn't create a partition by default, so you'll need to do that separately.

For this setup, the hardware test bed I will use is a PIC16F1503-based controller board left over from a recent automotive consulting project (**Figure 4**).

In the first step, as usual, choose the *File New Project* option to generate a project for the XC8 compiler. Next, start the MCC to generate a basic project skeleton. In this description, I assume you are already familiar with PIC programming, so I won't get into general handling in detail.

Not for Windows 7

It's important to be aware that Microchip explicitly advises against using the fifth version of their programming devices with Windows 7. I exclusively used machines running Windows 10 or 11 for this article. There are, however, some rumors circulating on the internet that generally claim it "works," at least when the programming device can regularly be connected to a Windows 10 or Windows 11 computer, where MPLAB performs various firmware upgrades and housekeeping tasks as part of a test delivery

The only crucial point is that two GPIO pins need to be declared as outputs. In the following steps, I will define pins RC4 and RC5 as outputs and then generate the code.

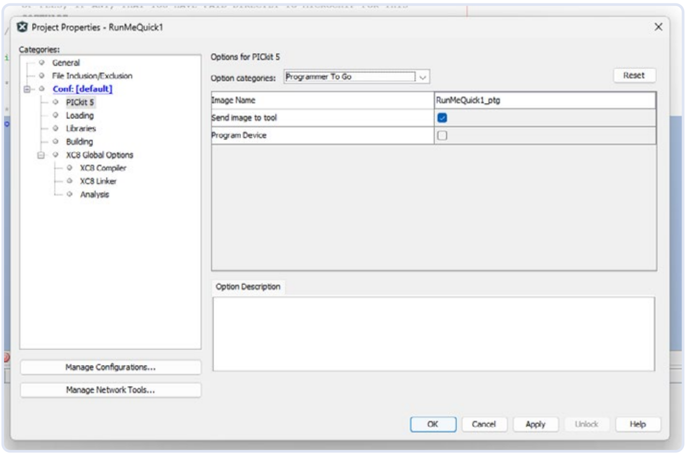
To make it easier to see the differences, I will write a program following a specific pattern within this example project skeleton, referred to as *RunMeQuick1*. This program will switch the two LEDs together at a relatively slow rate.

```
void main(void)
{
    // initialize the device
    SYSTEM_Initialize();

    // When using interrupts, you need to
  // set the Global and Peripheral Interrupt Enable bits
    // Use the following macros to:

    // Enable the Global Interrupts
    //INTERRUPT_GlobalInterruptEnable();

    . . .
    IO_RC4_SetHigh()  ;
    IO_RC5_SetHigh()  ;
    while (1)
    {
            IO_RC4_Toggle()  ;
            IO_RC5_Toggle()  ;
            __delay_ms(1000);
    }
}
```

In the next step, open the *Project Properties* view, where you'll find the settings for configuring the *Programmer-To-Go* operation, as shown in **Figure 5**.

The most crucial aspect here is the *Image Name* section, which is the name MPLAB assigns to the written-out file. The *Send image to Tool* option instructs the IDE to load the generated image onto the microSD card of the connected programming device.

Checking the *Program Device* checkbox is optional; it determines whether, in addition to the "actual" *Programmer-To-Go* function, a connected PIC microcontroller should be programmed with the provided image.



Options for PICkit 5

Option categories: Programmer To Go

Image Name	RunMeQuick1_ptg
Send image to tool	✓
Program Device	☐

Figure 5: MPLAB assists developers by configuring the **Programmer-To-Go** mode.

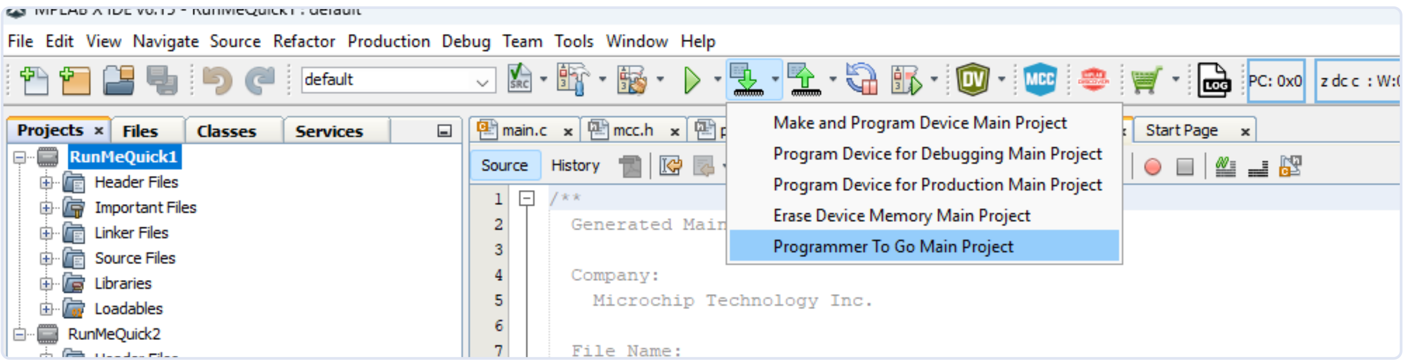
Using this option can make sense, for example, when you want to update both the *Programmer-To-Go* image and load current firmware onto an evaluation board for a final test.

The actual deployment must then be carried out — absolutely — through the menu shown in **Figure 6**, which initiates an "update" to the *Programmer-To-Go* image.

During deployment, sometimes an error message appears along the lines of `Transmission on endpoint 2 failed (err = -109)`. In my case, this issue was almost always resolved by restarting the PC. By the way, if you've just installed MPLAB, it's also recommended to reboot at this point. The deployment is complete when the status console displays the message `The debug tool is in programmer to go mode`.

Now we will return to the MPLAB start page in this next step to create another — almost identical — project, named *RunMeQuick2*. This will help illustrate the improvements in the second generation of PICkit 5. First, we will need to start the MCC again. The code intended to control the two signal LEDs now looks like this:

```
void main(void)
{
    . . .

    IO_RC4_SetLow()  ;
    IO_RC5_SetHigh()  ;
    while (1)
```



MPLAB X IDE v6.15 - RunMeQuick1 : default
File Edit View Navigate Source Refactor Production Debug Team Tools Window Help
default
PC: 0x0
z dc c : W:
Projects
Files
Classes
Services
RunMeQuick1
Header Files
Important Files
Linker Files
Source Files
Libraries
Loadables
RunMeQuick2
main.c
mcc.h
Source
History
Start Page
Make and Program Device Main Project
Program Device for Debugging Main Project
Program Device for Production Main Project
Erase Device Memory Main Project
Programmer To Go Main Project

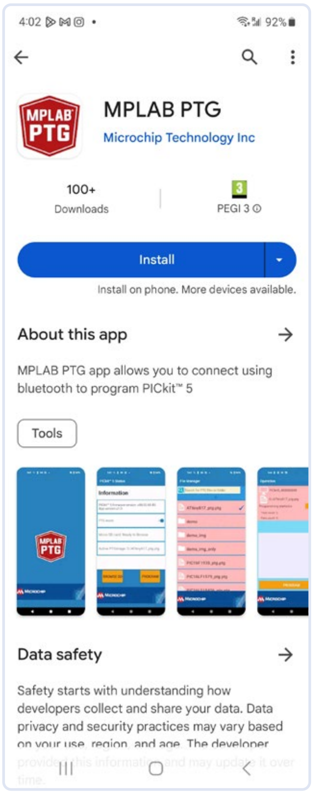
```
/**
  Generated Main

  Company:
    Microchip Technology Inc.

  File Name:
```

Figure 6: This entry transfers the image to the SD card.

```
        {
                IO_RC4_Toggle()  ;
                IO_RC5_Toggle()  ;
                __delay_ms(1000);
        }
}
```

To configure the connected PICkit, logically, you'll need to perform another execution, which should also conclude with the status message `The debug tool is in programmer to go mode`.

In the next step, you can disconnect the programmer from your workstation. I used a regular phone charger to provide power to the PICkit at this point. The flashing bright-green LED indicates that the device is in *Programmer-To-Go* mode, waiting for instructions.


MPLAB PTG

Microchip Technology Inc

100+ Downloads

PEGI 3

Install on phone. More devices available.

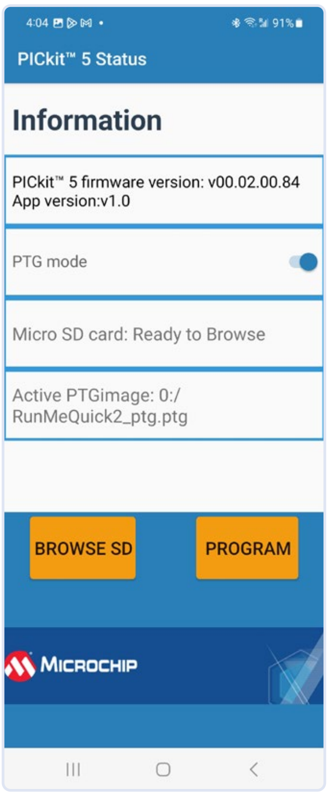
About this app

MPLAB PTG app allows you to connect using bluetooth to program PICkit™ 5

Data safety

Safety starts with understanding how developers collect and share your data. Data privacy and security practices may vary based on your use, region, and age. The developer provides this information and may update it over time.

Figure 7: This app simplifies remote programming.


PICkit™ 5 Status

Information

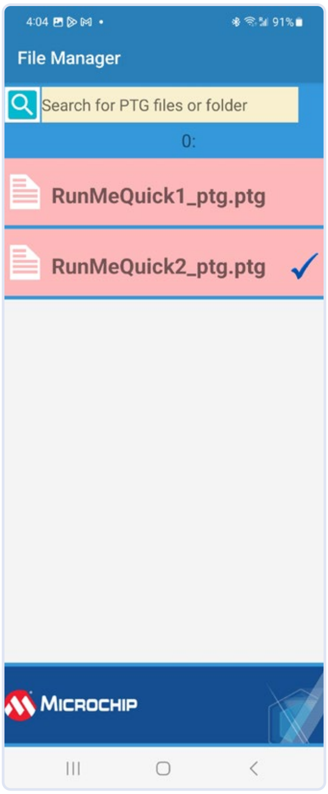
PICkit™ 5 firmware version: v00.02.00.84
App version:v1.0

PTG mode

Micro SD card: Ready to Browse

Active PTGimage: 0:/ RunMeQuick2_ptg.ptg

Figure 8: The start menu…


File Manager

Search for PTG files or folder

0:

RunMeQuick1_ptg.ptg

RunMeQuick2_ptg.ptg ✓

Figure 9: …and a selection of deployable images.

Now, we can open Android's Play Store or Apple's App Store to download the new control application, which is only compatible with the Bluetooth module of the PICkit 5. In the following steps, I will use a Samsung smartphone; the application appears in the Play Store as shown in **Figure 7**.

When using Bluetooth LE, note that the program will request various permissions during its initial launch — this is necessary due to Google's requirements, and is not critical or avoidable in any way.

In the next step, the scanning dialog appears, listing all the devices found in the vicinity — in my case, while carrying out the tests, I found it was necessary to sometimes click the *Cancel* button before I could access the list of found programming devices.

After completing the tasks, the system presents the screens shown in **Figures 8** and **9**, allowing you to directly deploy the desired firmware to connected target devices.

These days, the smartphone is the most common device we use that runs a familiar GUI. Running the application on such a device gives newbies to the application a head start compared to those starting from scratch working with the often more complex Project Explorer in MPLAB on a desktop machine. It's worth noting that a smartphone is almost always within reach, so that the (relatively more powerful) workstation is really no longer needed — especially in large deployments, this can significantly reduce costs.

Upgrading My Development Environment

The next experiment involves returning to my Windows 10 workstation, which I used, among other things, to write the textbook referenced in the box below. By default, MPLAB version 5.45 is installed on this machine, which I normally use for my commercial activities.

Now, it's time to pay another visit to the MPLAB website, where I have downloaded version 6.15. The latest version of MPLAB can be installed directly over any existing version.

After successfully completing the installation process, MPLAB 6.15 offers to use the settings from the previous version as the basis for configuring the fresh installation. As always, you'll need to confirm various settings in Windows Firewall. The actual IDE starts after downloading the *Microchip Offline Help* component.

Upon the initial launch of the IDE, an extensive parsing process follows, which seeks to bring all project files up to date, among other tasks. Additionally, it updates various caches created in MPLAB X and populates them with information from the projects.

For the next step, I chose to install demo example *CH9-Demo1* and connected the board, familiar from my textbook, to the programmer and the PC. Compilation was successful, but actual firmware deployment sometimes fails on Windows 10 with the previously mentioned error, `Transmission on endpoint 2 failed (err = -109)`. Apart from this, the new version of PICkit 5 seamlessly integrates into my existing development workflow and, thanks to optimizations, code deployment is, in fact, often faster in many cases.

Measure the Flow Using the ICD5

Microchip has been enhancing its MPLAB IDE with various convenient additional features for some time, aiming to make it easier for developers to visualize measurement and tracking data generated by the embedded application.

One of the interesting features of the ICD 5 is that it can provide this data-visualization engine from MPLAB with information about the power consumption of the connected application circuit.

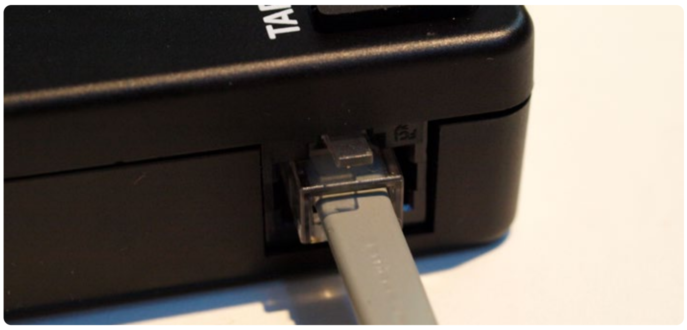


Figure 10: A minor inconvenience when using the ICD 5.

Table 1: The power monitor accuracy, according to Microchip.

Current and Voltage	Resolution	Full Scale
Current	29 µA / step	1.0 A
Voltage	0.2087 mV / step	6.8 V

The specification indicates a current measurement resolution of 0.29 µA — more measurement information for the power monitor is given in **Table 1**.

Now we can click on *Tools Plug-in Downloads* and choose *Microchip Plug-in Manager* in the window that pops up. Check if *MPLAB Data Visualizer* is already listed in the *Installed* section. This will typically be the case with a "fresh" installation.

Logically, you'll need to power the target hardware using the ICD5's built-in power source. The connector used in the ICD 5 is an RJ45 type, while older adapters such as the ICD 3 have an RJ11 cable. As shown in **Figure 10**, the connector makes good contact in principle, but it's not the most robust or secure of connectors

To save time and effort, we can use the LED examples created earlier for the PICkit to test the current measurement feature of the ICD 5. Disconnect the PICkit and connect the ICD 5 — for the sake of convenience, I will avoid the network cable together with its TCP/IP features and will instead connect via a USB-C cable.

Note that during the "initial" setup of an ICD, MPLAB needs to perform a firmware update for the FPGA — this one-time process takes some time.

Interestingly, this process often stalls at 93% and then continues "as usual" (**Figure 11**) — why MPLAB never reports reaching a 100% update status here is not entirely clear.

In the next step, click on *Window Debugging Data Visualizer* to start the *Data Visualizer* welcome screen. Then, in the *Power* section, click the *Play* button to begin data acquisition.

MPLAB responds by displaying the window shown in **Figure 12**, indicating there that no visualization form has been specified.

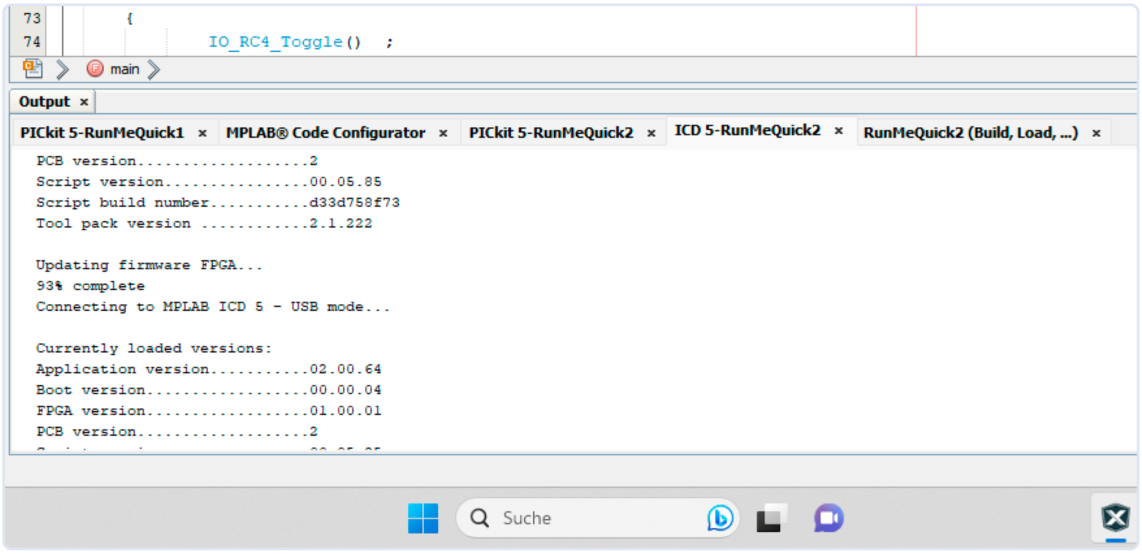
In response, click the *Plot Raw* function, which now shows a chart. If you run the program that alternates blinking between both LEDs, you'll see information about power consumption, as shown in **Figure 13**.

More interesting results can be obtained by using *RunMeQuick1* — this example turns both LEDs on and off alternately (**Figure 14**)

To Sum Up

Microchip has introduced a number of detailed improvements with the PICkit 5 and ICD 5, which will assist developers in both the debugging and "serial production" phase of a project's life cycle. Search engines for electronic parts such as *oemsecrets.com* are able to identify the best real-time pricing & stock levels of components from various distributors. The MPLAB PICkit 5 In-Circuit Debugger/Programmer retails at € 86, while the MPLAB ICD 5 In-Circuit Debugger/Programmer is available for € 360 — in no way too expensive for what is offered. Users of the PICkit 4 who are getting frustrated with the "old" interface may wish to upgrade to the new variant to cut down on the cable count. The power analysis feature is also very valuable and can potentially save significant costs when compared with an SMU or GPIB card.

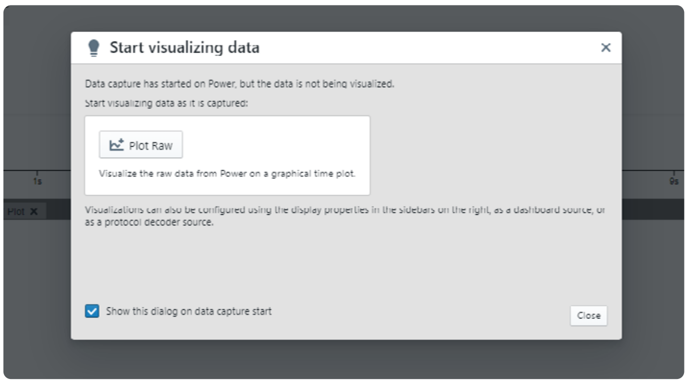
230571-01



```
    {
        IO_RC4_Toggle()   ;
```

```
PCB version...................2
Script version................00.05.85
Script build number...........d33d758f73
Tool pack version ............2.1.222

Updating firmware FPGA...
93% complete
Connecting to MPLAB ICD 5 - USB mode...

Currently loaded versions:
Application version...........02.00.64
Boot version..................00.00.04
FPGA version..................01.00.01
PCB version...................2
```

Figure 11: 93% indicates that the task is fully completed, here!



Start visualizing data

Data capture has started on Power, but the data is not being visualized.

Start visualizing data as it is captured:

Plot Raw

Visualize the raw data from Power on a graphical time plot.

Visualizations can also be configured using the display properties in the sidebars on the right, as a dashboard source, or as a protocol decoder source.

[✓] Show this dialog on data capture start

Close

Figure 12: MPLAB requests how the captured data should be represented or visualized.

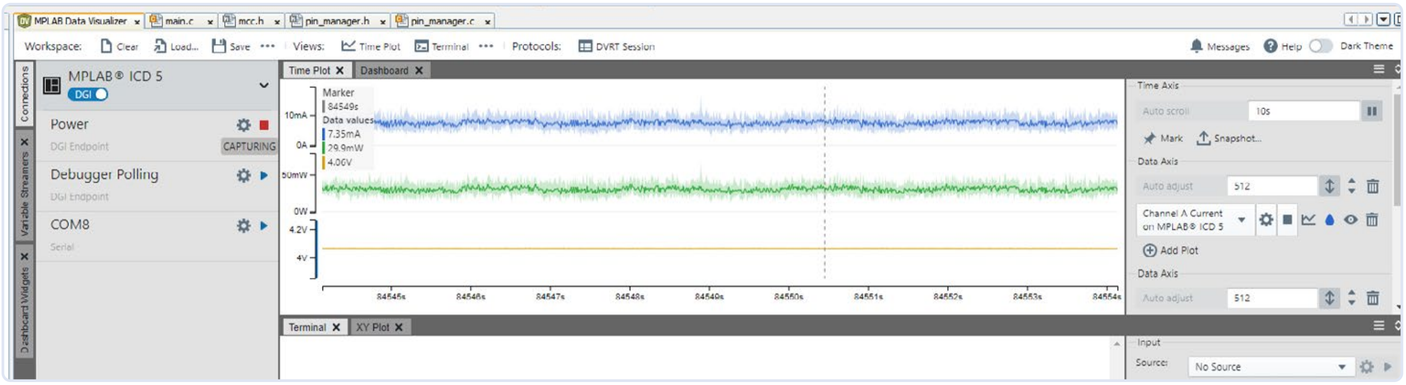


Figure 13: Here, there is always one LED on.

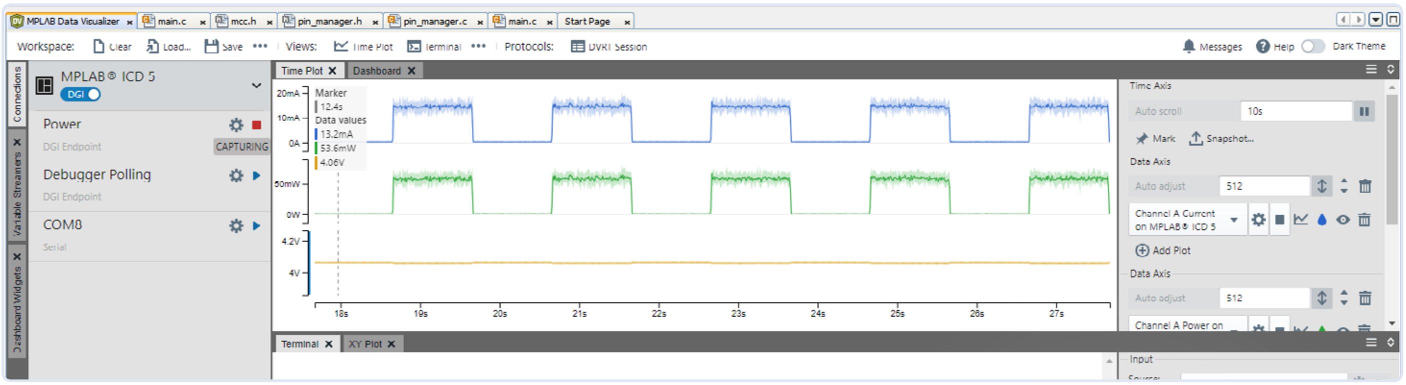


Figure 14: The example clearly shows the changes in current consumption.

Request your 15% discount code for these tools today!

Submit your contact details, select which tool you would like to get a 15% off discount voucher for and you'll receive a unique coupon code to redeem at MicrochipDirect.

https://page.microchip.com/pic5_icd5

WEB LINKS

[1] MPLAB PICkit 5 from Microchip: https://microchip.com/en-us/development-tool/pg164150

[2] MPLAB ICD5 from Microchip: https://microchip.com/en-us/development-tool/dv164055

[3] Application Note AN4121: https://microchip.com/en-us/application-notes/an4121